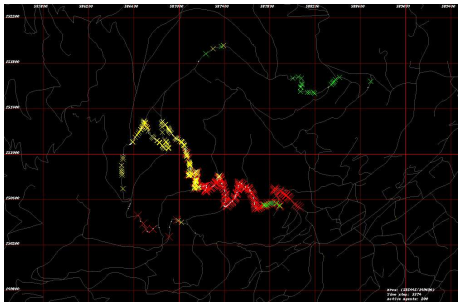
Eidgenössische Technische Hochschule
Swiss Federal Institute of Technology
Zürich

Department of Computer Science
Simulation group

Master Thesis 12. März 2004

Mental Maps for mobility simulations of agents



Daniel Kistler
(D-INFK)

Supervisor:
Prof. Kai Nagel,
Christian Gloor, Duncan Cavens

Prof.Kai Nagel
Simulation Group
Department of Computer Science
ETH Zürich

Inhaltsverzeichnis

Abstract

Mobilen Agenten, die in einer räumlichen und zeitlichen Umgebung agieren, benutzen in bisherigen Simulation meist globales Wissen um sich zu orientieren und den Tag zu planen. Dies entspricht jedoch nicht der Natur eines Individuums. Vielmehr spielt das lokale Umfeld des Agenten und die darin gewonnen Informationen eine zentrale Bedeutung. Die eigens erfahrenen räumliche Information speichert das individuum in einer mentalen Karte. Dabei stellen sich verschiedene Fragen: 'Wie wird das räumliche Wissen aus den wahrgenommen Informationen aufgebaut?', 'Wie beeinflusst dieses Wissen die Aktivitäten des Individuums?', 'Wie entsteht dadurch ein individuelles Verhalten?' und 'Was sind die Bedingungen für einen Wissensaustausch?'
Diese und andere Aspekten von Individuen mit einer lokalen Wahrnehmung, sowie die Umsetzbarkeit als Softwareagenten in einer Computersimulation werden untersucht.

Kapitel 1

Einführung

In einer vorgegebenen Umgebung mit räumlicher und zeitlicher Dimension sollen sich Individuen aufgrund der selbst gesammelten Erfahrung orientieren. Durch die so entstehende individuelle Wissensbasis wird das Verhalten des einzelnen Individuums stark geprägt. Dabei stellen sich folgende Fragen:

- Wie wird das Wissen aus den wahrgenommenen Informationen aufgebaut?
- Wie beeinflusst das individuelle Wissen die zukünftigen Aktivitäten?
- Auf welche Art entsteht das individuelle Verhalten?
- Welche Funktion und Einfluss hat dabei die vernetzte Struktur des Gehirns?
- Was sind die Bedingungen für einen Wissensaustausch zwischen Individuen?

Ausgehend von diesen Fragestellungen soll eine Simulation entwickelt werden, die diese Aspekte untersucht.

1.1 Die Karte im Kopf

Ein Lebewesen orientiert sich mit Hilfe einer persönlichen mentalen Karte in einer vertrauten Umgebung. Die mentale Karte repräsentiert die dem Individuum bisher bekannte Welt. Durch lokale Wahrnehmung wird diese ständig ausgebaut und weiter vervollständigt. Der Begriff Karte erinnert dabei meist an eine globale, metrikerhaltende geographische Karte, in der relative Distanzen und Richtungen einzelner Objekte zueinander einfach abzuschätzen sind. Ein Individuum nimmt jedoch nur in seiner lokalen Umgebung aus einer bestimmten Perspektive wahr. Dies erschwert die relative Beurteilung verschiedener Objekte zueinander enorm, da dem Individuum die objektive Distanz zu den einzelnen Objekten fehlt.
Trotzdem muss das Wissen, dass ein Individuum speichert, von ähnlicher Form sein zu dem Wissen, dass durch eine geographische Karte dargestellt wird, da das Verhalten in der räumlichen Orientierung beim Verwenden von globalen Information und lokalen Informationen sehr ähnlich ist. Dieses Verhalten betrifft nur die 'input/output' Relation und hat nichts gemein mit der Repräsentation des geographischen Wissens in der mentalen Karte oder auf einer Landkarte. Eine solche funktionale Ähnlichkeit darf jedoch kein Hinweis sein, dass in der mentalen Karte des Gehirns die 2-dimensionalen räumlichen und metrischen Verhältnisse erhalten bleiben.
Durch Analysieren von Skizzen von bekannten geographischen Umgebungen kann man im Selbstexperiment feststellen, dass eine gewisse zweidimensionale Struktur in der mentalen Karte erhalten bleibt, jedoch in verzerrter Form. Die Metrik in der Kartenwiedergabe

ist ungenau. Dies lässt darauf schliessen, dass die mentale Karte Metriken nur beschränkt wiedergeben kann. Trotzdem bleibt die räumlich topologische Struktur invariant, d.h. die relative Anordnung der einzelnen Objekte untereinander bleibt erhalten. Diese, kombiniert mit der sehr lokalen und subjektiven Wahrnehmung, lässt darauf schliessen, dass sich das Individuum mehr anhand lokalen topologischen Informationen orientiert als an metrischen Informationen. Metrische Informationen werden möglicherweise unabhängig und aufgrund von Erfahrungswerten separat gespeichert. Man kann sich z.B. vorstellen, dass die Zeit, kombiniert mit einer ungefähr individuell geschätzten Geschwindigkeit, ein bedeutender Faktor spielt, um sich eine Distanz sich zu merken.
Ein Individuum benützt räumliche Informationen hauptsächlich zur Navigation in seiner Umwelt und sammelt gleichzeitig neue Informationen. Dies lässt die Vermutung zu, dass unser räumliches Wissen auf Informationen, wie wir durch unsere Umwelt navigieren, basiert (Kuipers 'Map in the head' metaphor [10]).
Die zwei-dimensionale Struktur der Informationen, die das Gehirn wiedergeben kann, ist jedoch sehr lokal. Es wäre falsch anzunehmen, dass die gesamten im Gehirn gespeicherten räumlichen Informationen global referenziert werden können (Kuipers [10]). Sehrwohl kann ein Individuum lokal diese Informationen räumlich einigermassen korrekt wiedergeben, jedoch kommt es oft vor, dass diese lokalen Informationen nicht untereinander räumlich assoziiert werden können. Dies kann man sich einfach durch ein kleines Experiment veranschaulichen: Man nehme zwei bekannte Objekte in unterschiedlichen Städten, die man besucht hat. Jetzt soll man probieren, als Beobachter des einen Objektes die Richtung des anderen abzuschätzen. Man stellt fest, dass diese Aufgabe fast unmöglich zu lösen ist, benutzt man nur die Informationen, an die man sich erinnern kann.
Es ist zu vermuten, dass die einzelnen lokalen Karten nicht bedingt eine metrisches Referenz zueinander haben können. Da ein Individuum nur bewusst zwischen den einzelnen Orten navigieren kann, muss bekannt sein, wie man zwischen diesen einzelnen lokalen Karten navigiert. Konkret heisst das, dass wir zwar wissen, wie wir von einem Ort zum anderen gelangen, dabei jedoch unsere geographische Orientierung verlieren. Die Objekte werden in der mentalen Karte trotzdem zusammengehalten.
Bis jetzt wurde angenommen, dass Informationen symmetrisch abgespeichert werden, d.h. man nimmt an, dass die Beziehung von einem Ort A mit einem Ort B gleich mit einem Ort A mit einem Ort B ist. Dies ist jedoch nicht immer der Fall (Lee [17]). Vielmehr müssen Informationen asymmetrisch zur Richtung in der Mental Map vorhanden sein. Erfahrungswerte werden offensichtlich nicht aufgrund der verfügbaren geographischen Informationen, sondern eher subjektiv und lokal aufgrund von aufeinanderfolgenden Perspektiven des Individuums gespeichert (Kuipers [10]).

1.2 Prinzipien beim Simulieren von 'Der Karte im Kopf'

Ein Modell, dass die bisherigen diskutierten Prinzipien recht gut abdeckt, wurde bereits 1982 von B. Kuipers in seinem Artikel 'The map in the head metaphor' vorgeschlagen (Kuipers [10]). Dabei merkt sich das Individuum jeweils verschiedene Ansichten, sogenannte 'views', der momentanen Umgebung. Eine Ansicht muss dabei nicht visuellen Ursprungs sein, ist aber mit einer bestimmten Blickrichtung, d.h. Orientierung, verbunden. Durch geordnetes Verknüpfen von aufeinanderfolgend wahrgenommenen Ansichten durch mit Aktionen entstehen Routen. Orte sind dabei definiert als Ansichten, die durch Aktionen, die eine Rotation beschreiben, verknüpft sind, während Strassen durch Ansichten definiert sind, die durch Aktionen, die eine Vorwärtsbewegung beschreiben, verknüpft sind.
Durch das Aufsummieren dieser Routen entsteht eine Karte mit definierten Orten (Ansichten durch Rotation verbunden) und asymmetrischen Verbindungen (Ansichten die durch eine Vorwährtsbewegung verbunden sind). Die stärke des Modells liegt in seiner Einfachheit, Allgemeinheit und der Ausbaufähigkeit.

1.3 Information aquisieren

Wohl eines der grössten Probleme neben dem Modell der Wahrnehmung und des Abspeichern von Informationen ist die Frage, wie das Individuum die zur Wiederverwendung relevanten Informationen auswählt.
In einer ausführlichen Studie zeigte Yarbus [12] (1967), dass die Wahrnehmung einer komplexen visuellen Szene ein kompliziertes Muster von Fixationspunkten beinhaltet. Beim Betrachten einer Szene wird das Auge jeweils für kurze Zeit still gehalten und sucht sich dann ruckartig einen neuen Fixationspunkt. Der Bobachter ist sich normalerweise dieses Musters nicht bewusst, wenn er eine visuelle Szene (Ansicht) wahrnimmt. Die Frage, die sich dabei stellt, ist: Nach welchen Kriterien selektieren wir die verschiedenen Fixationspunkte, d.h. wie wählen wir die einzelnen Objekte in einer visuellen Szene aus?
Generell wird angenommen, dass der Selektionsprozess aus zwei Stufen besteht. Zuerst werden mit Hilfe von einer Art 'Preprocessing' alle interessanten Regionen extrahiert. Die gefundenen Gebiete, mit vermuteter höherer Relevanz werden dann sequentiell durch Fixation mit den Augen weiter untersucht (Theeuwes [9] (1993)). Was sich genau gemerkt wird, z.B. geometrische Objekte, Kanten, Farben etc. und wie dies mit einer semantischen Bedeutung in Verbindung gebracht wird, scheint noch weitgehend Thema der Forschung zu sein.
Betrachten wir diese Selektion von einem höheren Level der Beschreibung. Kahneman [3] klassifiziert Augenbewegungen in drei generelle Typen der Wahrnehmung, die sich unterscheiden durch die Situation, in der sich das Individuum befindet:

- **Spontane Wahrnehmung** kommt vor, wenn das Individuum eine Szene wahrnimmt, ohne eine spezifische Aufgabe im Hinterkopf zu haben. Was das Individuum genau wahrnimmt wird beeinflusst durch die Strukturen der Szene, die am meisten Informationen erhalten. Man kann sich natürlich nun fragen, wie diese Informationen wahrgenommen werden. Eine genaue Analyse dieser Frage geht jedoch über diese Arbeit hinaus. Wir nehmen deshalb an, dass dies von inidviduellen Präferenzen, sowie offensichtliche Kriterien an ein wahrgenommenes Objekt, wie z.B. Sichtbarkeit etc. abhängig ist.

- **Aufgabenbedingte Wahrnehmung** wird ausgeführt, wenn der Beobachter die Szene mit einer bestimmten Aufgabe im Hinterkopf betrachtet. Es wird nach bestimmten Informationen gesucht, die zur Lösung der Aufgabe beitragen können. Das kann zum Beispiel das Suchen eines bestimmten Objektes sein, oder das Verknüpfen einer Entscheidung, sowie des Zustands des Agenten mit relevanten Objekten. Wiederum nehmen wir an, dass offensichtliche Kriterien wie Sichtbarkeit etc. eine wichtige Rolle spielen bei der Analyse der Szene. Je nach Aufgabe haben Präferenzen des Individuums einen verschieden starken Einfluss auf das Resultat.

- Wenn wir uns **inneren Gedanken** zuwenden, ist unsere Aufmerksamkeit nicht auf die visuelle Szene konzentriert. Oft probiert das Individuum mit den Augen den Gedanken zu folgen, nicht jedoch Objekten der visuellen Szene. Es Resultiert eine gewisse Unaufmerksamkeit gegenüber äusseren Umwelteinflüssen.

Dass das Muster der Augenbewegung von der momentanen Gedankenaktivität beeinflusst wird, zeigt Abbildung 1.1. Da das Auge nur in Blickrichtung genau Wahrnehmen kann, muss das Muster aus den verschiedenen Fixationspunkten einer visuellen Szene sequentiell zusammengestellt werden. Man kann vermuten, dass es so etwas wie ein Gedächtnis geben muss, in dem die einzelnen Fixationspunkte einer visuellen Szene zwischengespeichert werden. Es ist jedoch unklar was genau in dem Gedächtnis gespeichert wird. Sind es bereits klassifizierte Informationen, oder nur Merkmale. Wir nehmen ersteres in der Simulation an.
In der Literatur spricht man auch von 'Ionic Memory', dass das gerade Gesehene beinhaltet

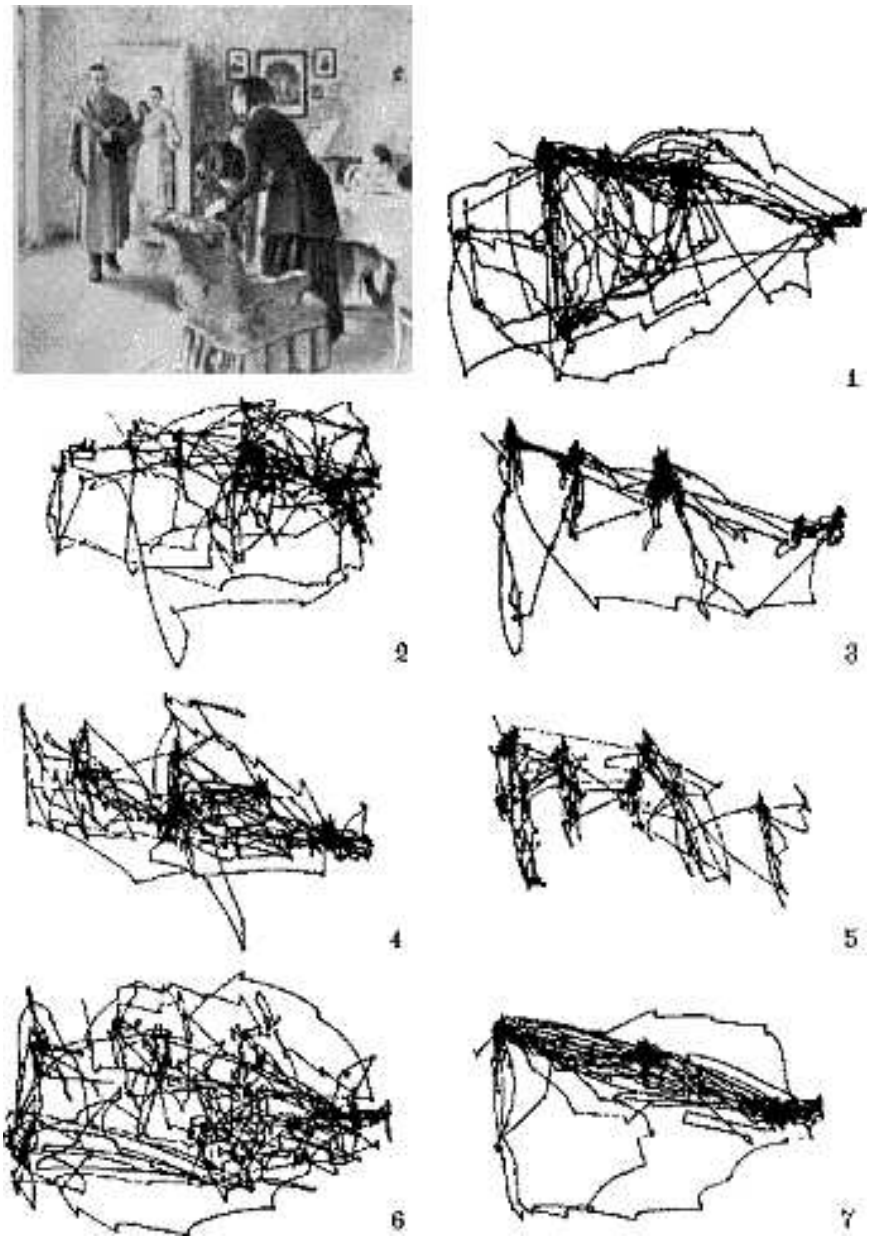


Abbildung 1.1: Die Grafik wiedergibt Aufzeichnungen der Augenbewegungen einer Versuchsperson (Yarbus [12] (1967)). Jede Aufzeichnung dauerte 3 Minuten. Der Versuchsperson wurden zum Analysieren eins Bildes (oben links) jeweils folgende Aufgaben gestellt: 1) keine Aufgabe (spontane Wahrnehmung), 2) schätze die materiellen Umstände der Familie ab, 3) schätze das Alter der Leute, 4) Vermute mit was die Familie beschäftigt war vor dem unerwartetem Besuch, 5) erinnere welche Kleider die Leute auf dem Bild trugen, 6) erinnere Dich an die Position der Leute und Objekte im Raum, 7) wie lange war der unerwartete Besucher weg von der Familie. Man sieht deutlich, dass sich das Auge sehr selektiv zu den gefragten Objekten und relativ zu deren Position bewegt.

(entdeckt von George Sperling 1956). Diese zwischengespeicherten Szenen wollen wir mit den in Section 1.2 vorgestellten 'views' in Verbindung bringen. Informationen bleiben im 'ionic memory' nur über eine sehr kurze Zeit erhalten, d.h. weniger als eine Sekunde.
Zusätzlich existierten bekannte Gedächtnistypen wie Kurzzeitgedächtnis ('short term memory' (STM)), das normalerweise 5-10 'Einträge' beinhaltet, sowie Langzeitgedächtnis ('long term memory' (LTM)). Das Kurzzeitgedächtnis soll Informationen für kurze Zeit präsent halten, während das Langzeitgedächtnis verantwortlich ist für die permanente Speicherung von Daten. Die Kapazität des Langzeitgedächtnisses ist weitgehend unbekannt.
Beim Kurzzeitgedächtnis sowie beim Langzeitgedächtnis gilt das 'power law of forgetting' (Wixted und Ebbesen [4] (1991), Rubin und Wenzel [1] (1996)). Dieses Gesetz sagt voraus, dass das Muster des Vergessens einer 'power' Funktion folgt, die von folgender Form ist:

$$f(t) = a * t^{-b}, \tag{1.1}$$

wobei t für Zeit steht und a, b reelle positive Zahlen sind.
Die Vergessensfunktion fällt im Langzeitgedächtnis viel langsamer ab, als im Kurzzeitgedächtnis. Es sei vorwegenommen, dass in der Simulation die Vergessensfunktion für das Langzeitgedächtnis deshalb vernachlässigt wird.
Die Frage ist, wie Kurzzeitgedächtnis, Langzeitgedächtnis und 'ionic memory' miteinander verknüpft sind? In der Simulation nehmen wir an, dass klassifizierte Objekte in einem Kurzzeitgedächtnis zwischengespeichert und über eine bestimmte Zeit präsent bleiben. Das 'ionic memory' existiert parallel zum Kurzzeitgedächtnis. Ist ein wahrgenommenen Objekt noch genügend präsent im Kurzzeitgedächtnis zur zeit der Wahrnehmung, gelangt es ins 'ionic memory'. Ist die Analyse einer Szene beendet, gelangt die zwischengespeichert Ansicht, für die permanente Speicherung ins Langzeitgedächtnis.

Visuelle Adaption Es wird vermutet, dass der Kontext der Objekte eine wichtige Rolle einnimmt im 'preprocessing'. Der Kontext beschreibt das zeitlich nur langsam veränderbare Muster, den Hintergrund einer visuellen Szene. Ein Objekt ist erst erkennbar, wenn es sich genügend stark vom gegenwärtigen Kontext abhebt.
Wenn man zum Beispiel über eine Graslandschaft spaziert, erregt ein Baum mit grösserer Wahrscheinlichkeit die Aufmerksamkeit, als wenn man sich im Wald befindet, da der Kontext und das Objekt gut differenzierbar sind. Solche Adaptionseffekte sind ein bekanntes Phänomen in der menschlichen Wahrnehmung.
Die Fragen, die man sich stellt, sind: 'Wie sieht eine solche Adaptationskurve bezüglich der Adaptionszeit aus?' und 'Wie ist ein Kontext aus einer lokalen Perspektive definiert?'.
Wir nehmen an, dass der Kontext heterogen sein kann, d.h. aus verschiedenen Objekttypen zusammengesetzt. Die Länge der Zeitspanne, in der das Individuum das Objekt wahrnimmt, beschreibt, wie stark das Objekt dem Kontext zugewiesen wird. Die Objekte in einer visuellen Szene sollen unabhängig voneinander wahrgenommen werden. Die Assimilationskurve, die beschreibt, wie ein Objekt in den Kontext übergeht, soll eine von der Zeit abhängige, nichtlineare Funktion sein. Sie hat die Eigenschaft, dass das Objekt anfangs schnell an Attraktivität für das Auge verliert.
Es ist anzumerken, dass der Kontext selbst eine wichtige räumliche Bedeutung haben kann. Diese ist jedoch nicht zur lokalen Navigation geeignet, sondern abstrahiert die Umgebung auf einem höheren Level. Mehr dazu in Sektion 1.5.

1.4 Speicherstrukturen eines Individuums

Wie wird das Wissen in den verschiedenen Speichertypen eines Individuums abgelegt? Grundsätzlich ist bekannt, dass die Speicherstruktur eines Individuums sich aus einem Netz

von Neuronen zusammensetzt. Dabei existieren verschiedene Prozessebenen, die das Wissen aus der wahrgenommenen Szene extrahieren und schliesslich im visuellen Zentrum ('visual cortex') des Gehirns verarbeiten. Man kann mit Experimenten zeigen, dass je nach fokusiertem Objekt unterschiedliche Regionen des visuellen Zentrums aktiv werden. (z.B. Riesenhuber M., Poggio T. [16](2000)).
Eine stimulierte Region des Gehirns deutet jedoch noch keinesfalls darauf hin, dass wir uns auch der Wahrnehmung bewusst werden. Man kann sich vorstellen, dass dieses Bewusstwerden einerseits von dem Zustand des Individuums abhängt (vgl. Sektion 1.3), anderseits von der Stärke des Stimulus, welche durch die momentane Signalstärke und der Rückkopplung ('feedback') alter Signale bestimmt wird.
Das Neuronennetz des visuellen Zentrums mit den dazugehörigen Vorverarbeitungsregionen wie die Retina des Auges, LGN (lateral geniculate nucleus), stellt einerseits eine Erkennungsfunktion für wahrgenommene Objekte dar, hat anderseits durch die objektspezifische lokale Aktivität gewisser Regionen zugleich eine Speicherfunktionalität. Der ganze Mechanismus ist sehr komplex und wird nur ansatzweise verstanden. Es scheint, dass 'Objekt identifizieren' und 'im Speicher lokalisieren' sehr nahe beieinander liegt.
Die vernetzte Struktur des Gehirns ist nicht rein zufällig, sie wiederspiegelt sich ziemlich stark in unserem Wissen wieder, indem Informationen aufgrund von Erfahrungswerten miteinander assoziiert werden. Erinnert man sich z.B. an einen Löwen, denken wir sofort auch an Käfig, Zirkus oder Savanne. Mit anderen Worten, Informationen werden oft in einem Kontext von anderen Informationen wiedergegeben (Verhalten beschrieben von z.B. Mandler G. [7] (1980).
In Betracht solcher Verknüpfung von erfahrungsbedingt verwandten Informationen wird klar, wie anpassungsfähig diese Neuronenstruktur ist. Jedes Individuum verknüpft diese Informationen anders, da es eine unterschiedliche Wahrnehmung hat, eventuell bedingt durch seine Präferenzen.
Die Frage, die sich stellt, ist: Aufgrund von welchen Kriterien wird eine Szene widererkannt? Wenn man sich vor Augen führt, wie eine Szene gelernt wird (Sektion 1.3), ist es naheliegend zu vermuten, dass eine Erkennung aufgrund der enthaltenen Objekte erfolgt. Dies würde eine hierarchische Struktur von Informationen nahe legen, wobei abstrakte Objekte durch einfachere Objekte zusammengesetzt werden (vgl. Sektion 1.5).

1.5 Hierarchisieren von Informationen

Eine der wohlgrössten Stärken der menschlichen Wahrnehmung ist das enorm schnelle Einordnen von wahrgenommenen Informationen. In nur einem Bruchteil einer Sekunde kann ein Mensch ein Objekt aus der visuell wahrgenommenen Szene erkennen. Dabei erkennt er nicht nur einzelne Objekte, sondern hierarchisiert diese auch gleichzeitig mit Hilfe von verschiedenen Methoden (Hirtle und Joinides [8]).

1.5.1 Typen von Hierarchien

Menschen konstruieren Hierarchien durch Abstrahieren von Informationen. Sie bilden geordnete Klassen von Informationen. Der Mensch verwendet dabei verschiedene Hierarchien, ohne das er sich bewusst ist dies zu tun. In der Literatur unterscheidet man grundsätzlich zwischen drei Typen von Hierarchien, die durch die Funktion, die sie erzeugen, unterschieden werden [15]:

- Aggregation
- Generalisation
- Filterung

Aggregation Man spricht von Aggregationshierarchie, falls Mengen von Objekten zu neuen Objekten zusammengeführt werden. Die Objekte werden dabei gemäss einem bestimmten Attribut, wie z.B. eine kleine gegenseitige Distanz, zu Objekten höherer Abstraktion in der Hierarchie zusammengeführt. Die Aggregation ist der am Häufigsten verwendete Hierarchietyp.

Generalisation Die Generalisationshierarchie definiert Klassen als generischer, je höher sie in der Hierarchie sind. So gehören z.B. 'Häuser und Fabriken' zur generischen Klasse Gebäude. Die Regeln, die entscheiden, welche Klasse zu welcher generischen Klasse gehört, sind vorgegeben. Die darauf aufbauende Generalisationsfunktion sagt also explizit, welche Klassen zu welchen generischen Klassen generalisieren. Die generischste, d.h. allgemeinste Klasse, ist zuoberst in der Hierarchie.
Generalisation basiert in der untersten Stufe auf einer Klassifikation. Einem Objekt muss zuerst eine semantische Bedeutung zugewiesen werden, damit die Generalisierungsregeln darauf angewandt werden kann. Als Beispiel betrachte man das Gebäude 'Zürcher Hauptbahnhof', dass zuerst als Bahnhof klassifiziert werden muss. Durch anwenden von einer Generalisierungsregel kann dann ausgesagt werden, dass ein Bahnhof auch ein Gebäude ist.

Filerhierarchie Die Filterhierarchie wendet eine Filterfunktion auf ein Set von Objekten an. Dabei entsteht eine Untermenge, die höher in der Hierarchie ist. Die Objekte auf einer höheren Hierarchiestufe findet man auch in den abgeleiteten tieferen Hierarchiestufen wieder.

Es ist zu bemerken, dass alle Hierarchien in einer einzigen Obermenge enden. Die drei Typen von Hierarchien hängen voneinander ab. So kann die Filterhierarchie aus der Generalisationshierarchie abgeleitet werden, und die Aggregationshierarchie entsteht aus der Filterhierarchie.
Die Verwendung von Hierarchien hat den Vorteil, dass Wissen schnell auf eher handhabbare Portionen reduziert werden kann. Dabei kann die Verarbeitungszeit von Aufgaben auf den Informationen stark verringert werden.

Anmerkung: Kommunikation Hierarchisierungsregeln werden nicht nur durch direkte Erfahrung von Informationen geformt. Indirekte Erfahrung via z.B. Kommunikation mit einem anderen Individuum oder Vererbung beschleunigen den Prozess. Dadurch entwickelt sich eine gemeinsame Basis des Verständnisses.

1.5.2 Hierarchien für die räumliche Orientierung

Hierarchien werden von Menschen für das Abstrahieren von verschiedensten Informationen gebraucht. Für das räumliche Wissen bildet die Generalisation von semantischem Wissen und die räumliche Aggregation vom Individuum wahrgenommenen Objekte wohl die wichtigsten Hierarchien.
Ein Individuum unterscheidet wahrgenommene Merkmale eines Objektes folgendermassen:

- Die Information des Objektes dient zur Wiedererkennung (subjektive Bedingung)
- Die Information des Objektes dient zur Klassifikation (objektive Bedingung)

Informationen, die zur Wiedererkennung eines ganz spezifischen Objekts dienen, können als Identität des Objektes aufgefasst werden. Haben Objekte gemeinsame Merkmale, werden diese aggregiert und bilden eine Klasse. Objekte können durch Klassifikation aufgrund verschiedener Merkmale (was einer Filterung entspricht) in Gruppen eingeteilt werden.

Werden diese Merkmale abgeschwächt, spricht man von einer generischeren Klasse. Diese ist durch Klassifikation mit den abgeschwächten Merkmalsbedingungen entstanden.
Ein Objekt kann heterogener oder homogener Natur sein. Homogene Objekte sind räumlich verbundene Einheiten, also z.B. eine Eiche, ein Baum, eine Pflanze, etc. Sie sind definiert durch eine Aggreagation von Erkennungsmerkmalen, die zur Klassifikation verwendet werden. Heterogene Objekte sind nicht räumlich verbundenen Einheiten, d.h. ein Wald, besteht z.B. aus Bäumen, Rehen, Pilzen, etc.. Man spricht auch von einer Ansammlung von Objekten. Heterogene Objekte sind definiert durch eine Aggregation von Objekten homogener oder heterogener Natur. Sie dienen als Merkmale und werden zur Klassifikation verwendet. Die einzelnen Objekte erhalten mit der Klassifikation einen Namen, die semantische Bedeutung eines Objektes. Diese Benennung von Objekten ist insofern an eine objektive Bezeichnung geknüpft, wenn eine gemeinsame Ebene des Verstehens vorhanden sein muss unter den Individuen. Anderseits ist die Identifikation jedem Individuum selbst überlassen und daher subjektiv.
Homogene Objekte sind bidirektional in einer Hierarchie verbunden. Durch eine Abschwächung der Klassifikationsmerkmale kann auf ein Objekt das höher in der Hierarchie, d.h. generischer ist, geschlossen werden. Umgekehrt kann durch Hinzufügen von Klassifikationsmerkmalen auf Objekte die tiefer in der Hierarchie liegen, d.h. spezifischer sind, geschlossen werden. Ein Beispiel ist: (Eiche, Baum, Pflanze, etc.).
Heterogene Objekte sind unidirektional in einer Hierarchie verbunden. Die Merkmale selbst sind Objekte. Durch Weglassen von Merkmalen kann nicht auf ein neues Objekt geschlossen werden. Jedoch kann man durch die Häufigkeit oder des Vorkommens eines Merkmalobjektes auf ein Objekt höher in der Hierarchie schliessen. Man spricht z.B. von Wald, wenn der Baumanteil hoch ist, oder man Objekte findet, die nur im Zusammenhang mit Wald vorkommen.
Der Unterschied zwischen räumlichen und semantischen Hierarchien wird in (D.Maio S.Rizzi (1996) [14]) beschrieben (Abbildung 1.2). Der Begriff räumlich wird dort jedoch unklar dargestellt. Er wird nicht nur für die gegenseitige Nähe von Objekten verwendet, sondern bezieht sich auch auf vom Menschen festgelegte Grenzen, wie z.B. einzelne Stadtbezirke. Vordefinierte Informationen solcher Art sind oft schwierig in die individuelle Wahrnehmung zu integrieren, da sie für eher für Informationsquellen ausgelegt sind, wie sie z.B. auf einer Landkarte verwendet werden. In [13] stellen Maio und Rizzi einen 'cluster'-Algorithmus vor, der dynamisch, also parallel zur Wissensaquisation, eine distanzbasierende Aggregationshierarchie aufbaut. Das Individuum kann Distanzen normalerweise nur sehr ungenau abschätzen. Vielmehr spielt die relative Distanz zwischen den Objekten eine Rolle. Objekte, die durch eine kleine relative Distanzen verbunden sind, werden oft als ein einziges Objekt wahrgenommen.

1.6 Erweiterung der 'Mental Map' mit einer zeitlichen Dimension

Bis jetzt sind wir davon ausgegangen, dass die Welt vor allem aus statischen Komponenten besteht. Die Realität ist jedoch, dass die Welt auch dynamische Komponenten beinhaltet, die das räumliche Wissen mitbeinflussen können. Mit 'dynamisch' ist die zeitliche Zustandsabhängigkeit gemeint. Man kann folgende grobe Unterscheidung für physisch existente und wahrnehmbare Objekte machen:

- **Statische Objekte (räumlich invariant)** sind räumlich invariant, d.h. können regelmässig am gleichen Ort wieder aufgefunden werden. Statische Objekte sind vor allem als Orientierungsmarken nützlich.

- **Willkürliche dynamische Objekte** Willkührliche dynamische Objekte sind deshalb schwierig zu behandeln, da sie einerseits durch ihre räumliche Dynamik nur sehr

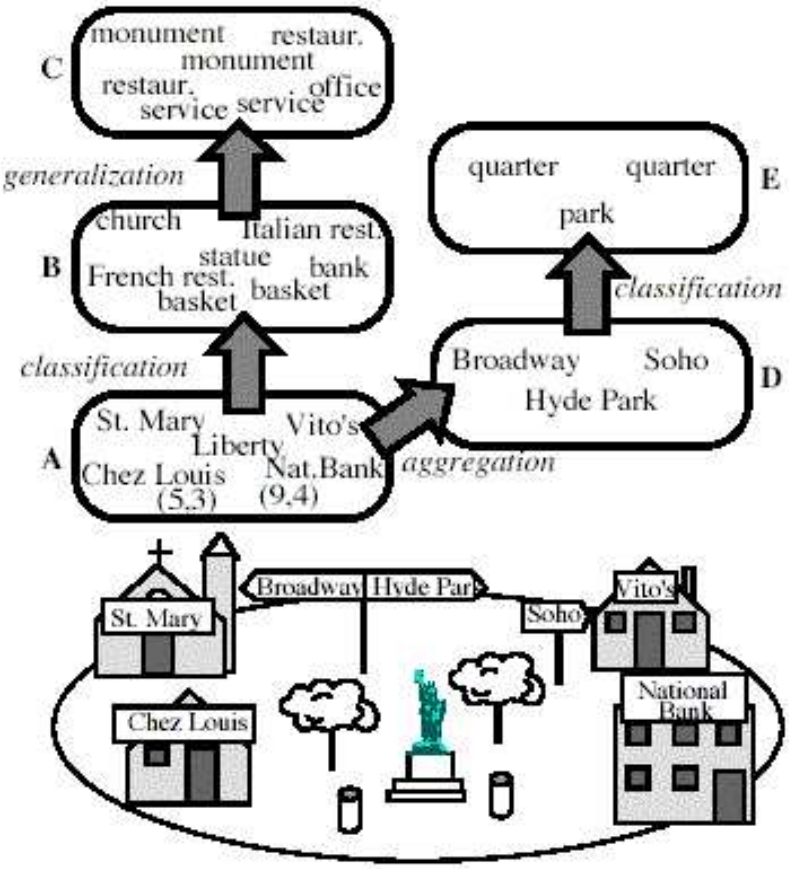


Abbildung 1.2: In der Abbildung ist das Hierarchisierungsmodell von Maio und Rizzi in [14] gegeben. Es werden Generalisations- und Aggregationshierarchien verwendet für die räumliche Umgebung. Die Grafik veranschaulicht ebenso, dass die Aggregatonskriterien sehr unklar sind. In [13] verwenden Maio und Rizzi Distanzen, um Informationen zu aggregieren ('clustern' und darauf eine Hierarchie aufzubauen.

beschränkt vorhersehbar, anderseits oft nicht kontrollierbar sind und kein Reaktionsbewusstsein haben. Ein Beispiel ist das Wetter.

- **Konstante dynamische Objekte** Konstante dynamische Objekte sind gekennzeichnet durch nach einem bestimmten Muster auftretende Ereignisse. Diese Muster können entstehen durch die Beeinflussung anderer dynamischen Ereignissen, die einem konstanten Muster folgen. Beispiele für solche Verhaltensmuster sind z.B. dass es am Morgen viel Verkehr hat, da sehr viele Leute zur Arbeit fahren, oder dass am Mittag die meisten Restaurants besetzt sind.

Verhaltensmuster von dynamischen Objekten können sich negativ oder positiv auf die Attraktivität von Objekten bezüglich den persönlichen Absichten eines Individuums auswirken. Die Definition eines Individuums beinhaltet das eigene Verhaltensmuster ,nach den persönlichen Bedrfnissen selbständing zu optimieren. Es ist deshalb sinnvoll, dass das statisches und dynamisches Wissen mit in das eigene Wissen eingebaut wird. Durch eine Erweiterung des räumlichen Wissens um eine zeitliche Dimension werden konstante dynamische Muster zu statischen Mustern. Willkürliche dynamische Objekte bleiben nur schwer fassbar.

1.7 Bewertung von Informationen

Ein Individuum bewertet aufgrund von Erfahrung das gesammelte räumliche Wissen. Die Bewertung wiedergibt die Attraktivität eines Objektes im Bezug auf eine bestimmte Aufgabe. Die Attraktivität von Objekten wird durch individuell wahrgenommene Informationen beeinflusst, die normalerweise durch das Objekt selbst aktiviert wurde.
Eine Bewertung versucht grundsätzlich ein bestimmtes Muster von Attraktivitäten festzuhalten, die auf das Verhalten eines Agenten Einfluss haben. Durch das Betrachten von zusätzlich einer zeitlichen Dimension werden auch konstant dynamische Objekte durch ein statisches Muster fassbar, und können in die Wissensbasis des Individuums einfliessen. Willkührliche Dynamik ist dagegen nur schwer fassbar, da sie keinem bestimmten zeitlichen und räumlichen Muster folgt. (vgl. Sektion 1.6). Die Auswirkungen auf eine bestimmte Attraktivität sind jedoch vorhersehbar. Dieses Muster auf allgemeinerer Ebene ist abschätzbar. So beeinflusst z.B. Regen die schöne Aussicht immer in einer ähnlichen Form. Man stellt fest, dass das Bewerten von Informationen nur schwer fassbar und komplex ist. Ist eine Bewertung optimal für eine Aufgabe, muss sie es nicht sein für eine andere. Bewertungsmuster sollten deshalb idealerweise für einen bestimmten Aufgabenraum definiert sein.

1.8 Verwenden von Informationen der 'Mental Map'

Die Verwendungsart von Informationen eines Individuums hat einen starken Einfluss auf die Informationsselektion (vgl.Sektion 1.3), sowie die Datenstruktur (vgl. Sektion 1.4). Sind diese auf die Anforderungen eines Individuums optimiert, so wirkt sich dies positiv auf die Extraktionseffizient der Daten aus dem Speicher aus. Der Grund ist, dass einerseits unnütze Information gar nicht vorhanden ist, und anderseits Information gemäss Erfahrungsmustern verknüpft ist. Es wird erwartet, dass so die Komplexität beim Lösen von einer Aufgabe durch das Individuum stark reduziert wird.
Erstaunlich ist die Grösse eines Aufgabenraums, den ein Inidividuum nach diesen Prinzipien bewältigen kann. Trotzdem sind Unterschiede bedingt durch die Spezialisierung eines Lebewesens vorhanden.
Die bisher untersuchten räumlichen und zeitlichen Informationen werden vor allem für die Orientierung und Planung von Aktivitäten eines Individuums verwendet. Dabei sollen die gespeicherten Informationen Hinweise, wie gut sie Erfahrungsgemäss für eine bestimmte Aufgabe geeignet ist, beinhalten. Bei der Wissensaquisation sollen also nicht nur räumliche und zeitliche Fakten gespeichert, sondern auch aufgrund des Empfinden des Individuums bewertet werden. Jedes Individuum hat eine persönliche Grundempfindung, sogenannte Präferenzen. Dieses Wissen beeinträchtigt die Planung des Individuums und auf etwas abstakterer Ebene sein Reaktionsverhalten.(vgl.Sekton 1.7). Das Reaktionsverhalten beruht dabei auf unerwartete Bewertungsähnderungen, die durch willkührliche dynamische Ereignisse hervorgerufen werden (vgl. Sektion 1.6). Die bisherige Aufgabe verliert durch die erneute Bewertung mit der geänderten Werten an Bedeutung. Ist die Bewertungsabweichung gross, wäre es für das Individuum von Vorteil, sich mit einer neuen, den Gegebenheiten angepassten Aufgabe auseinanderzusetzen.
Geht man davon aus, dass mehrere Individuen in einer offenen Gemeinschaft leben, d.h. miteinander kommunizieren können, so kann die Information, die in einem Individuum gespeichert ist, übergreiffend verwendet werden. Die Voraussetzung dafür ist jedoch, dass eine gemeinsame, semantische Informationsbasis vorhanden ist. Mit anderen Worten muss die Bedeutung einzelner Informationsstücke für jedes Individuum gleich sein. Ist dies gewährleistet, kann die Information über die Individuengrenze hinaus verwendet werden. Andernfalls läuft der Agent in Gefahr, die Inforamtion falsch zu interpretieren. Nimmt man eine Gruppe von unabhängigen Individuen mit jeweils eigenem Charakter und damit auch eigenen Präferenzen an, beginnt die eindeutige Semantik bereits zu wackeln. Sind die Ab-

weichungen gering, kann mit grosser Wahrscheinlichkeit auch erwartet werden, dass die Auswirkungen ebenfalls gering sind. Ein Individuum sollte in jedem Fall den Wahrheitsgehalt einer Information pruefen und eventuell die Gewichtung der Kommunikationspartner dementsprechend anpassen. Durch Kommunikation von andern gelernte Informationen, mit Bedacht auf das räumliche und zeitliche Wissen in einer mentalen Karte, bedeutet, Wissen wahrzunehmen, ohne dieses direkt erfahren haben zu müssen.

Kapitel 2

Simulation einer räumlichen Mentalen Karte

Das Ziel dieses Abschnittes ist, Methoden zu entwickeln, die das in Kapitel 1 diskutierte Verhalten eines Individuums in einer Computersimulation umsetzen. In der Simulation wird das Individuum durch einen Softwareagenten repräsentiert. Ein Agent ist eine autonome Einheit, die ein eigenständiges Verhalten simuliert.

2.1 Die Wanderersimulation

Die Simulation baut auf der Wanderersimulation ALPSIM auf (vgl. Gloor Ch et al. [5]). Es soll das Verhalten von Wanderern auf authentischen GIS Daten der Region Gstaad in der Schweiz simuliert werden.

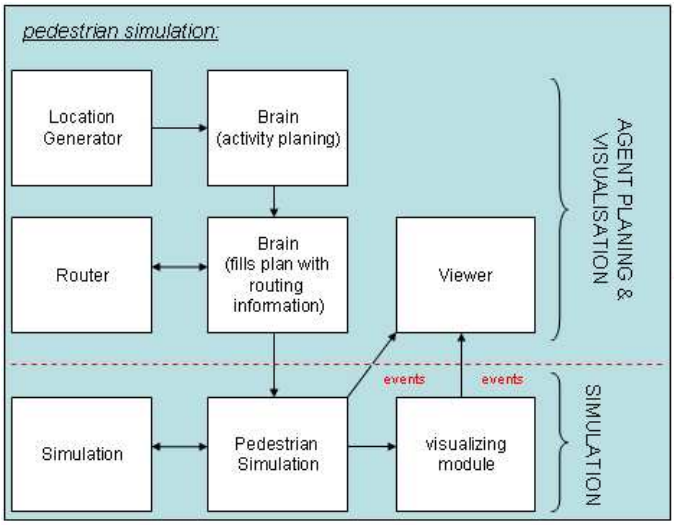


Abbildung 2.1: Die ursprüngliche Wanderersimulation soll so erweitert werden, dass jeder Agent ein eigenes räumliches Bewusstsein erhält. Die bisherige Simulation besteht zum Einen aus einem Teil, der die Simulation der Agenten und deren Visualsierung beinhaltet, und zum Andern aus einem Teil, der das räumliche Bewusstsein des Agenten koordiniert. Bisher teilen alle Agenten dasselbe räumliche Bewusstsein.

2.1.1 Aufbau der Simulation

Die Simulation ALPSIM gliedert sich in zwei Teile. Der eine ist verantwortlich für das Simulieren der Wandererumgebung, der Wahrnehmung und der Visualisierung der Agenten, der andere beschäftigt sich mit dem Planen von Tagesrouten, um bestimmte Aktivitäten auszuführen. Die Planung einer Route des Agenten basiert auf dem Gesamtwissen der simulierten Welt. Abbildung 2.1 gibt die Komponenten der bisherigen Simulation wieder.
In Kapitel 1 wird jedoch von Agenten ausgegangen, die als Individuum agieren und nur lokales, selbst wahrgenommenes Wissen besitzen. Die Simulation ALPSIM soll erweitert werden, dass jeder einzelne Agent eine individuelle Wissensbasis besitzt.
Die Simulationskomponenten kommunizieren via das 'Multicasting' von XML-Event-Strings. In den XML-Events sind interne Kontrollinformation der Simulation oder die momentane Wahrnehmung eines Agenten wiedergegeben. Abbildung 2.2 zeigt die XML-Event-Strings, die für die Agenten der erweiterten Simulation relevant sind.

a) Agentlinks:

```
<event type="location" time="577" id="35" locationtype="street" linkid="4371" action="exit" x="588295.924968"
                       y="150454.918673" />
<event type="location" time="577" id="35" locationtype="street" linkid="4289" action="enter" x="588295.924968"
                       y="150524.918673" />
```

b) Agent view events:

```
<event type="view" time ="342" agentID="1" agentXPos="8543343.3" agentYPos="8543443.3" viewDirection="45"
                               fieldofView="120"  objectID="45" viewAngle="33.5" distanceFromViewer="124"
                               viewPercent="12.3">
```

c) Agent feel events:

```
<event type="feel" feeltype="saevent0" time="1251" id="35" x="587783.289296" y="150560.518585"
                   objectID="street4" />
```

Abbildung 2.2: Vom Agenten werden drei Typen von XML-Event-Strings verwendet: a) 'location-events', welche angeben, wann der Agent ein Objekt betritt oder verlässt, b) 'view-events', die für jedes vom Agenten wahrgenommenen Objekt gesendet werden und c) 'feel events', welche eine emotionale Bewertung von visuellen Informationen bedeuten (z.B. schöne Aussicht). Weitere 'XML-Event-Strings' werden zur Kommunikation von Kontrollinformationen zwischen den einzelnen Komponenten der Simulation verwendet.

2.1.2 Hinzufügen von einem individuellen Bewusstsein

Das Individuelle Bewusstsein ist mehr als nur eine zusätzliche Karte für das einzelne Individuum. Sie beeinflusst folgende Komponenten eines Agenten:

- Die Wahrnehmung von Objekten (vgl. Sektion 1.3)
- Die Planung des Agenten (vgl. Sektion 1.8)
- Die Reaktionsfähigkeit des Agenten (vgl. Sektion 1.8)

Abbildung 2.3 gibt das revidierte Modell der Simulation grob wieder. Anstatt einer Einheit, die für alle Agenten Aktivitätenpläne generiert, soll diese Aufgabe jeder Agent selbst übernehmen. Mit anderen Worten, bisher haben die Agenten keine eigenständige Intelligenz, sie führen blind die Aktivitätenpläne aus die sie vom Planungsmodul (Brain) erhalten.
Für die individuelle Planung steht das persönlich erfahrene, räumliche Wissen des Agenten zur Verfügung. Ein Agent lässt sich somit folgendermassen grob skizzieren:
Jeder Agent besitzt eine Komponente, welche die individuellen Wahrnehmungsereignisse entgegennimmt und die darin enthaltene Information verarbeitet. Eine weitere Komponente bewertet diese Information und legt sie in einer mentalen Karte ab. Schliesslich wird aus

den gespeicherten Informationen in der generierten, mentalen Karte und einem vorgegebenen Generalisationswissen in einer letzten Komponente ein Aktivitätenplan für einen Tag generiert.
Die Simulation vervollständigt den Aktivitätenplan mit räumlichen Koordinateninformationen, damit er in der Wanderumgebung ausgeführt werden kann. Eine detaillierte Struktur der einzelnen Komponenten, die einen Agenten beschreiben, ist in Appendix B gegeben.

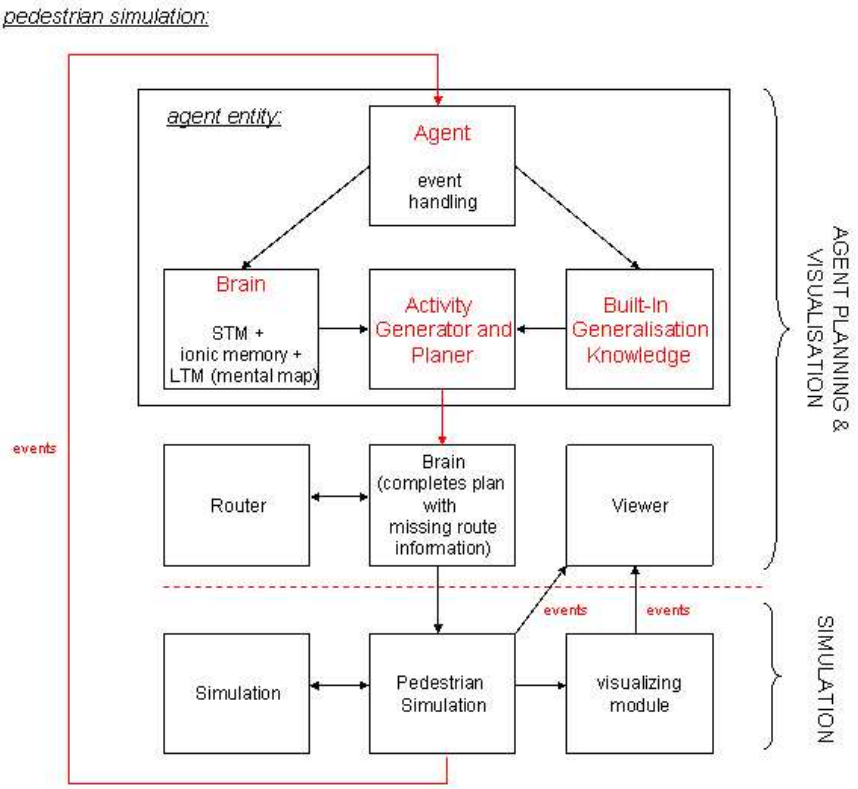


Abbildung 2.3: Die Abbildung zeigt in groben Zügen die Änderungen der bisherigen Simulation. Jeder Agent ist geprägt von einer Komponente, die die individuellen Wahrnehmungsereignisse des Agenten verarbeitet, einer Komponente die die mentale Karte baut, und zuguterletzt eine Komponente, die das individuelle Agentenwissen wiederverwendet, um Routen zu generieren, auf denen der Agent verschiedene Aktivitäten ausübt. Jeder Agent ist eine eigenständige Komponente, die mit der restlichen Simulation kommuniziert.

2.2 Speicherkomponenten und individuelle Wahrnehmung

Ein Individuum legt die wahrgenommenen visuellen Informationen in verschiedenen Speichern ab (vgl. Sektion 1.3). In der Simulation sind dies ein Kurzzeitgedächtnis(STM), ein Langzeitgedächtnis(LTM) und ein sogenanntes 'ionic memory'(IM). Alle wahrgenommenen visuellen Objekte werden im Kurzzeitgedächtnis zwischengespeichert. Ein Objekt ist eindeutig und wird aufgrund von Merkmalen wie z.B. Häufigkeit, der Distanz, etc. bewertet. Die Bewertung wird bei jeder zusätzlichen Wahrnehmung aktualisiert. Nimmt der Agent ein Objekt wahr und liegt die Bewertung im Kurzzeitgedächtnis über einem bestimmten Schwellenwert, so gelangt es ins 'ionic memory'. Das 'ionic memory' sammelt alle gleichzeitig wahrgenommenen Objekte, d.h. die momentane visuelle Ansicht des Agenten. Die komplette Ansicht wird dann in der Datenstruktur des Langzeitgedächtnis

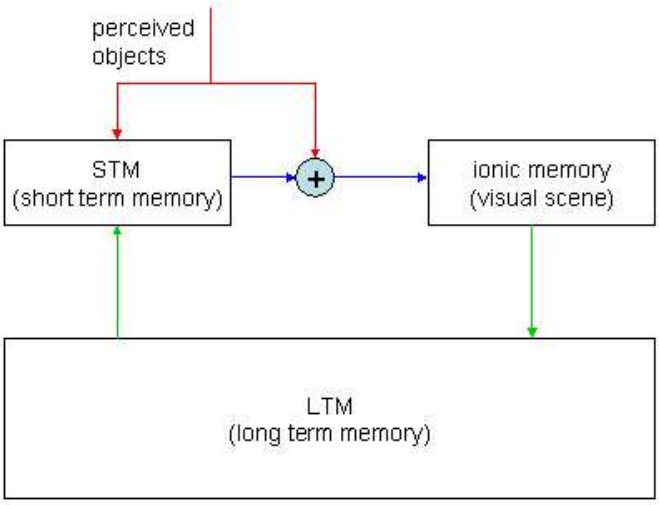


Abbildung 2.4: Die Speicherfähigkeit der Agenten in der Simulation besteht aus drei Komponenten: Dem Kurzzeitgedächtnis (STM), dem 'ionic memory', sowie dem Langzeitgedächtnis (LTM). Ein Objekt im Kurzzeigedächtnis ist eindeutig. Ihm wird eine Bewertung, aufgrund von wahrgenommenen Metriken wie z.B. Frequenz, Distanz, etc. zugewiesen. Die Bewertung wird mit jeder neuen Wahrnehmung des Objektes aktualisiert. Nimmt der Agent ein Objekt wahr und übersteigt die Bewertung des Objektes im Kurzzeitgedächtnis einen bestimmten Schwellenwert, so gelangt dieses ins 'ionic memory'. Im 'ionic memory' werden alle gleichzeitig wahrgenommen Objekte, d.h. die momentane visuelle Ansicht des Agenten, zwischengespeichert. Diese gelangt zur permanenten Speicherung ins Langzeitgedächtnis. Falls die Ansicht schon im Langzeitgedächtnis vorhanden ist, wird diese aktualisiert.

abgelegt. Falls der Eintrag bereits vorhanden ist, wird er aktualisiert. Die Datenstruktur ist speziell für die räumliche Wahrnehmung ausgelegt. Die Anordnung der Speicherkomponentnen ist in Abbildung 2.4 wiedergegeben.

2.2.1 Langzeitgedächtnis oder LTM (long term memory)

Das Langzeitgedchtnis hat die Aufgabe, Informationen permanent zu speichern. Seine genaue Kapazität ist nicht bekannt, sie ist jedoch physisch begrenzt. Nicht das Speichervolumen ist jedoch das wichtigste Kriterium eines guten Langzeitgedächtnisses, sondern dessen Extraktionsgeschwindigkeit der Informationen.
Beim sequentiellem Durchsuchen von Informationen und einem sehr hohen Speichervolumen ist die Antwortverzögerung gross. Eine Vernetzung der Informationen, die aufgrund von Erfahrungswerten miteinander verknüpft sind, kann die Verzögerung verkürzen (vgl. Sektion 1.4). Beim räumlichen Wissen, werden von Vorteil räumlich lokale Informationen assoziiert. Mit diesen Daten wird sich der Agent mit grosser Wahrscheinlichkeit als nächstes auseinandersetzen.
Die Struktur des Speichers in der mentalen Karte ist folgendermassen aufgebaut (vgl. Abbildung 2.5). Grundsätzlich nimmt ein Agent visuelle Objekte war. Ein Objekt hat dabei eine eindeutige Identität zur Identifikation und einen Typ, der dem Objekt eine semantische Bedeutung zuweist. Gleichzeitig wahrgenommene Objekte sind miteinander assoziiert und formen eine Ansicht. Der Agent merkt sich dabei die topologische Anordnung der verschiedenen Objekte, die ungefähre Distanz und die Signifikanz. Die Signifikanz ist die Bewertungsabweichung des Objektes vom Wahrnehmungsschwellenwert im Kurzzeitgedächtnis. Die schnelle Verknüpfung des Wissens wird erreicht, indem die jeweils zugehörigen Objekte einer Ansicht eine direkte Referenz auf die entsprechende Ansicht haben und umgekehrt. Ansichten sind miteinander verbunden, falls sie zeitlich aufeinanderfolgend wahrgenom-

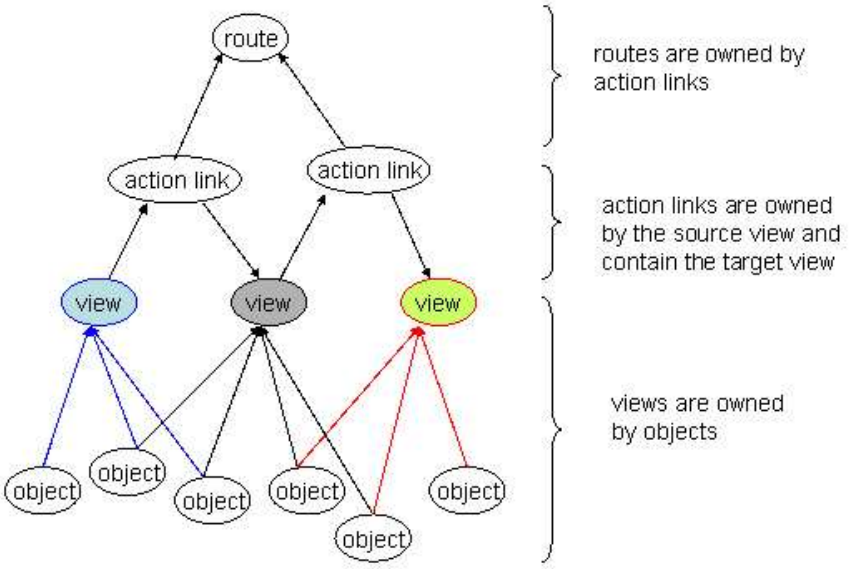


Abbildung 2.5: Der räumliche Langzeitspeicher ist hierarchisch aufgebaut. Die verschiedenen Abstraktionslevels sind: Objekte, Ansichten 'Action Links' und Routenobjekte. Eine Ansicht beinhaltet die gleichzeitig wahrgenommen Objekte. Ein 'Action Link' verbindet zwei Ansichten mit einer Aktion. Die Aktion führt den Agenten von der einen Ansicht in die andere über. Ein Routenobjekt, verbindet zwei Ansichten mit je einem Routenknoten als 'source', der auch eine Verzweigung ist. Die Objekte auf einer höheren Stufe der Hierarchie sind in den direkt abgeleiteten Objekten als Referenz enthalten. Diese bidirektionale Verknüpfung erhöht die Mobilität beim Suchen von Objekten.

men werden. Eine solche Verbindung definiert einen 'Action Link'. In einem 'Action Link' wird die durchschnittliche Zeit gespeichert, die der Agent benötigt, um die Aktion auszuführen. Die Aggregation aller 'Action Links' formt einen gerichteten Graphen (Abbildung 2.6). Diese Karte, assoziiert mit den ungefähren metrischen Erfahrungswerten eines Agenten, repräsentiert die Umgebung so, wie sie der Agent wahrnimmt. In der Simulation ist die Referenz des 'Action Link' an die Startansicht gebunden. Der Link selber hält Referenzen auf die Startansicht und Zielansicht.
In der Simulation existieren Objekte, die nicht punktuell begehbar sind. Dies sind vorläufig nur Strassenobjekte. Eine Strasse ist definiert als die Verbindung zweier Routenverzweigungen und ist mit einer eindeutigen Identität versehen. Sie kann verschiedene Ansichten besitzen, welche die gleiche Ursprungsidentität ('source') haben. Eine genaue Positionsidentifikation innerhalb des Objektes gibt es nicht. Nur die Reihenfolge, in der die einzelnen Ansichten wahrgenommen wurden, ist bekannt. Ein Strassenobjekt, zusammen mit den umschliessenden Routenverzweigungen, definiert ein Routenobjekt. Ein Routenobjekt ist charakterisiert durch eine Startansicht, eine Endansicht und den enthaltenen 'Action Links'. Die enthaltenen 'Action Links' enthalten eine Referenz auf das Routenobjekt.
Sämtliche punktuell begehbaren Objekte, die der Agent wahrnimmt, sind miteinander durch Routen verbunden. Der Agent nutzt die Routenobjekte zur Planung der Tagesrouten. Eine Planung basierend auf 'Action Links' wäre ebenfalls denkbar, jedoch aufwendiger. Die Voraussetzung ist, dass der Agent möglichst viele der Routenverzweigungen erkennt. In der Simulation wird dies garantiert, indem auf jeder Verzweigung mindestens eine Ansicht generiert wird. Zusätzlich wird die Wahrscheinlichkeit, Objekte zu erkennen, auf einem Verzweigungsobjekt erhöht, da ein gewisser Entscheidungsdruck vorhanden ist.
Es ist eine gemeinsame Ebene des Verständnisses notwendig, um zwischen den unterschiedlichen Simulationskomponenten Objekte zu kommunizieren. Mit anderen Worten,

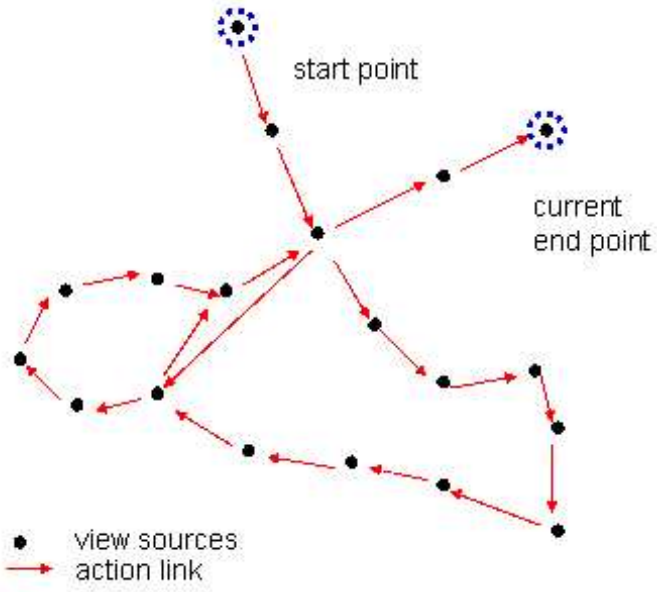


Abbildung 2.6: Die Aggregation der verschiedenen durch 'action links' verknüpften Ansichten des Agenten resultiert in einer räumlichen Karte. Verbindungsinformationen sind asymmetrisch. Dies bedeutet, dass um von einem Ort A nach B zu gelangen, nicht dieselben 'Action Links' verwendet werden, wie um von B nach A zu gelangen. Dies hat den Grund, dass unterschiedliche Objekte wahrgenommen werden, da der Agent eine unterschiedliche Blickrichtung hat. Ebenso kann nicht davon ausgegangen werden, dass andere Metriken gleich sind, wie z.B. die Zeit.

sämtliche Objekte, die der Agent wahrnimmt, sind auch der Simulation bekannt. Möchte der Agent interessante Orte als neue Objekte definieren, müsste das Wissen mit den restlichen Simulationskomponenten ausgetauscht werden. Dies ist jedoch nicht implementiert. Der Agent kann jedoch Koordinaten kommunizieren, die er besuchen soll. Dies würde die Generierung von internen Knoten ermöglichen.

Anmerkung: Von Individuen werden Objekte aufgrund von visuellen Merkmalen erkannt. Dies ist jedoch ein sehr komplexer und von der Wissenschaft noch nicht vollständig verstandener Vorgang. Bisherige Ansätze, solche Objekte aufgrund von Merkmalen zu klassifizieren und ihnen eine semantische Bedeutung zuzuweisen, sind weder robust noch effizient. Für die Simulation sind beide Kriterien jedoch von essentieller Bedeutung. Die Erkennung von 3D Objekten wird umgangen, indem für jedes Objekt eine eindeutige Identität eingeführt wird.
Der Nachteil von Objekterkennung via Identitäten ist, dass nur eine eindeutige Identifikation eines Objektes möglich ist. Die menschliche Wahrnehmung ist jedoch probabilistisch. Je mehr Merkmale für ein Objekt erkannt werden, desto eindeutiger kann es im Speicher identifiziert werden. Man spricht auch von einer Generalisierungshierarchie (vgl. Sektion 1.5). Die eindeutige Identität für ein Objekt steht dabei zuoberst in der Hierarchie. In der Simulation wird zusätzlich zur Identität ein Objekttyp eingeführt, der angibt, auf welcher Stufe der Generalisierungshierarchie das Objekt erkannt wird. Die Eindeutige Identität ist nötig damit das Objekt einfach im Speicher wiedergefunden werden kann.

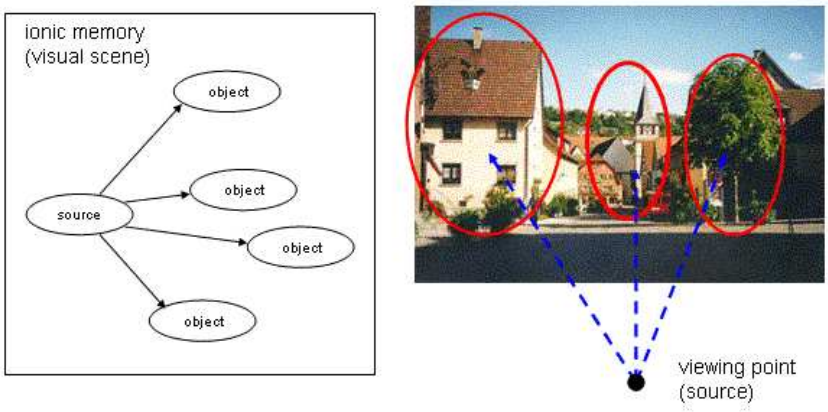


Abbildung 2.7: Im 'ionic memory' wird die momentane individuelle Ansicht einer Szene zwischengespeichert. Eine Ansicht umfasst alle gleichzeitig wahrgenommenen visuelle Objekte. Dabei kann man nur bedingt von 'gleichzeitig' sprechen. Das Individuum empfindet zwar die Szenenaufzeichnung als gleichzeitig, in Wirklichkeit jedoch verbringt es eine gewisse Zeit, mit Augenbewegungen die Szene unbewusst zu analisieren. In der Simulation gehen wir von Gleichzeitigkeit aus. Eine Ansicht besteht aus dem Quellenobjekt (von wo aus man betrachtet) und den Objekten (die man betrachtet).

2.2.2 Visuelle Momentanaufnahme im 'ionic memory'

Im 'ionic memeory' wird eine individuelle Momentanaufnahme einer visuellen Szene – eine Ansicht – zwischengespeichert (Abbildung 2.7, vgl. Sektion 1.2) . Es werden nur Objekte betrachtet, die bei ihrer Wahrnehmung im Kurzzeitgedächtnis eine Bewertung über einem erforderlichen Schwellenwert haben. Eine Ansicht sei definiert als ein Quellenobjekt, von dem Beobachtet wird, und den gleichzeitig wahrgenommenen Objekte in richtiger topologischen Anordnung. Von Gleichzeitigkeit kann man dabei nur bedingt sprechen. Das Individuum empfindet zwar die Szenenaufzeichnung als gleichzeitig, in Wirklichkeit verbringt es jedoch eine gewisse Zeit, mit Augenbewegungen die Szene unbewusst zu analysieren. In der Simulation gehen wir von Gleichzeitigkeit aus. Die entsprechenden wahrgenommenen Objekte sind mit einem Zeitstempel versehen. Mit topologischer Anordnung ist die relative Anordnung der Objekte zueinander bezüglich der Blickrichtung gemeint. Ein Beispiel für eine topologische Anordnung von einem Objekt A und einem Objekt B ist z.B. dass Objekt A links von Objekt B ist.
Der Agent in der Simulation analysiert in konstantem zeitlichen Abstand die Umgebung und erhält die entsprechenden 'events'. Je mehr dieser Daten in der mentalen Karte gespeichert werden, desto grösser ist einerseits der Speicherbedarf und anderseits der Aufwand, die entsprechenden Daten wiederzufinden. Deshalb sollen die Daten reduziert werden und nur die signifikanten Ansichten gespeichert werden. Eine Ansicht sei signifikant, sobald sie eine gewisse Änderung zur vorherigen Ansicht aufweist.

Zwei aufeinanderfolgende Ansichten werden nach folgenden Kriterien differenziert:

- Sourceobjekt
- Perspektive
- Ähnlichkeit der Objekte in korrekter topologischer Reihenfolge

Ausschlaggebend für einen solchen Vergleich ist die Anordnung und die Anzahl der gesehenen Objekte in einer Ansicht. Ähnlich definieren wir Ansichten, die ein identisches

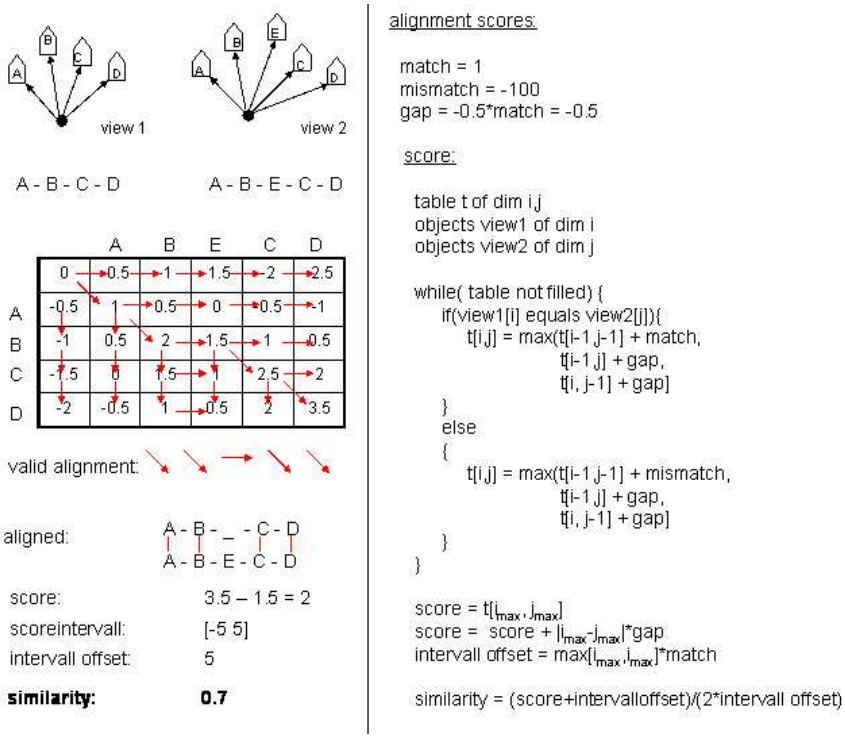


Abbildung 2.8: Der rechte Teil der Grafik veranschaulicht, wie zwei Objektsequenzen miteinander verglichen werden und ein Mass für deren Gleichheit berechnet wird (similarity). Man versucht, eine optimale Bewertungssumme für ein Alignment zu finden, indem man 'matches', 'mismatches' und 'gaps' bewertet. Die einzelnen Bewertungen sind so gewählt, dass die berechnete Ähnlichkeit den Wert 'eins' hat, falls die Objektfolgen identisch sind, und 'null', falls sie total verschieden sind. Total verschieden heisst hier, dass zwei Objektmengen die gleiche Anzahl, jedoch keine gleichen Elemente besitzt. Die roten Pfeile geben jeweils die Richtung an, von wo der maximale Wert für das nächste Tabellenfeld berechnet wurde. Auf der rechten Seite ist der Algorithmus zur Berechnung der einzelnen Werte gegeben.

Quellenobjekt haben. Ist die topologische Anordnung zweier Objekte in der Ansicht vertauscht, so nehmen wir eine unterschiedliche Perspektive an und generieren eine neue Ansicht ('view'). Ist die Anordnung erhalten, wird mit Hilfe von einem Alignment ermittelt, wie ähnlich die topologische Anordnung der gesehenen Objekte zu vergleichenden Ansichten sind. Konkret wird untersucht, wie viele Objekte einer Ansicht korrelieren, wie viele neue Objekte vorhanden und wie viele verschwunden sind. Dazu ist ein Bewertungsmass nötig. Es wird eine Form von dynamischem Programmieren (Needleman S.B. and Wunsch C.D. [2]) verwendet. Eine vollständig identische Ansicht wird mit dem Wert 'eins', eine total verschiedene Ansicht mit dem Wert 'null' bewertet (vgl. Abbildung 2.8). Überschreitet der Vergleichswert eine bestimmte Schwelle, nehmen wir an, dass die Ansichten identisch sind, andernfalls handelt es sich um eine neue Ansicht.

Mit den selben Kriterien wird geprüft, ob eine Ansicht bereits im Langzeitgedächtnis vorhanden ist. Findet man mehrere Kandidaten, die ähnlich sind und gleiche Perspektive haben, wird der ähnlichste gewählt und mit der neuen Ansicht ergänzt.

Der adaptive Schwellenwert entscheidet, ob ein Objekt in das 'ionic memory' gelangt. Er ist geprägt von den zuvor wahrgenommenen Objekten. Das bedeutet jedoch auch, dass eine Ansicht nicht immer gleich wahrgenommen werden muss. Es ist durchaus möglich, dass ein Objekt in einer Ansicht pötzlich fehlt. Durch das Verwenden des A"hnlichkeitskriteriums wird das Objekt mit hoher Wahrscheinlichkeit trotzdem wiedergefunden. Bei Ansichten mit nur sehr wenigen Objekten kann ein einziges Objekt entscheiden, ob eine Ansicht wahr-

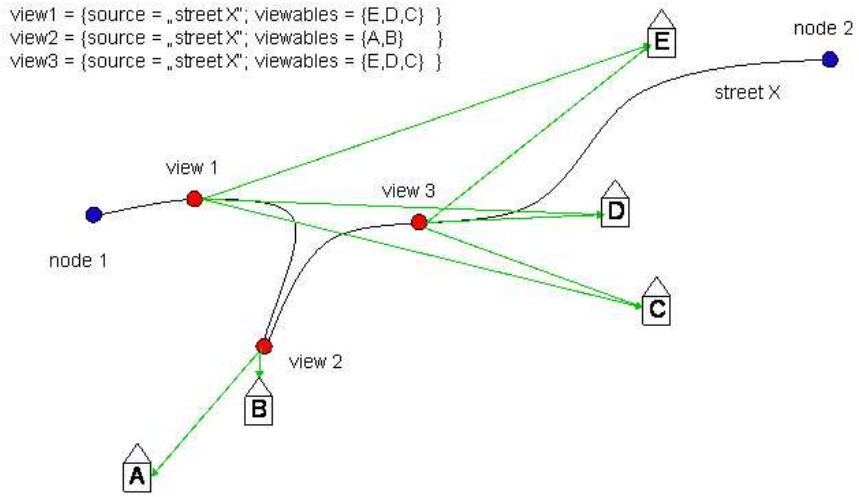


Abbildung 2.9: Die Abbildung zeigt ein Beispiel von drei Ansichten, die die StrasseX, als 'source' Objekt besitzen. Die erste Ansicht ('view1') und die letzte ('view2') sind dabei visuell nicht voneinander zu unterscheiden. Nimmt der Agent 'view2' wahr, ist dies für ihn dieselbe wie 'view1'. Merkt man sich die gerade wahrgenommenen Ansichten der aktuellen 'source', kann das Problem relativ einfach für ein neues Routenobjekt gelöst werden. Der Agent merkt, dass 'view1' zwar gleich aussieht wie 'view2', jedoch nicht die Gleiche sein kann, da sie bereits wahrgenommen wurde. Die Ansichten werden durch ihre Identität (Referenz) unterschieden. Versucht man später, die Ansicht wiederzufinden, ist das Resultat jedoch nicht mehr eindeutig. Die Extraktion beruht nur auf den visuellen Objekten einer Ansicht. Es müssen zusätzliche Informationen zugezogen werden.

genommen wird oder eben nicht. Unter Umständen dann das zu einer etwas instabilen Wiedererkennung führen. Dies betrifft vor allem in einem Umfeld mit nur wenigen visuellen Objekten.
Mit dem Ähnlichkeitsschwellenwert kann die Menge an Informationen, die ins Langzeitgedächtnis gelangen soll, kontrolliert werden. Gleichzeitig wird so die Toleranz beim Vergleichen der Ansichten kontrolliert.
Eine neue Ansicht wird in der Speicherstruktur des Langzeitgedächtnisses (LTM) abgelegt. Zuerst wird geprüft, ob die Ansicht bereits vorhanden ist. Durchaus können solche vorhanden sein, die identisch sind, jedoch an verschiedenen Orten auf dem selben 'source' Objekt vorkommen, wie in Abbildung 2.9 veranschaulicht wird. Dies kann zu Uneindeutigkeiten führen, falls eine Ansicht aus dem Langzeitgedächtnis extrahiert werden soll. Folgende Kriterien sollen bei der Extraktion aus dem Langzeitgedächtnis diese Fehlerquelle reduzieren:

- Die Ansicht darf nicht in der Menge der bereits wahrgenommenen Ansichten vom momentanen Routenobjekt mit derselben 'source' sein.

- Besteht immer noch eine Uneindeutigkeit, wird die Ansicht, die näher an der zuletzt wahrgenommenen des momentanen Routenobjekts ist, als gleich eingestuft. Die Ansicht darf einen bestimmten Abstand nicht überschreiten. Es ist unwahrscheinlich, dass die Ansicht gleich ist, falls z.B. 5 seit der zuletzt wahrgenommenen dazwischen liegen. Die Wahrscheinlichkeit ist sehr gering, dass ganze 5 aufeinanderfolgende Ansichten beim letzten Mal, als die Strasse abgelaufen wurde, nicht erkannt wurden. Die Ansicht wird sonst als neu eingestuft.

Mit den beiden Kriterien wird die Fehlerquote reduziert. Es ist jedoch nicht ausgeschlossen, dass die Uneindeutigkeit nicht korrekt gelöst wurde. Da die Reihenfolge der Ansichten mit

gleicher Source für die Anwendungen in diesem Projekt unbedeutend ist, wird nicht näher auf das Problem eingegangen.

Ergänzen einer Ansicht: Muss die Ansicht ergänzt werden, geschieht dies folgendermassen: Die gemäss Alignment korrespondierenden Objekte bleiben. Objekte, die neu dazugekommen sind, werden zusätzlich in die Ansicht eingefügt. Die topologischen Relationen müssen dabei erhalten bleiben. Ist ein Objekt in der neuen Ansicht nicht mehr vorhanden, wird es trotzdem in der Ansicht gelassen. Probleme tauchen auf, wenn Objekte, die eingefügt und gelöscht wurden, unmittelbar aufeinander folgen. Es sind zu wenige Informationen vorhanden, damit diese topologisch eindeutig zusammengefügt werden könnten (vgl. Abbildung 2.10). Es wird die Teilanordnung der Ansicht übernommen, die eine höhere Signifikanz hat. Die Signifikanz eines Objektes ist die durchschnittliche prozentuale Abweichung der Objektbewertungen des Kurzzeitgedächtnisses vom Schwellenwert, gemessen am Zeitpunkt, an dem die Ansicht jeweils wahrgenommen wurde (Gleichung 2.1).

$$significance = \frac{\sum_{i=1}^{n} \frac{score_i}{threshold_i}}{n}, \tag{2.1}$$

wobei die $significance$ die Abweichung vom Schwellenwert angibt, n die Anzahl der Wahrnehmungen der Ansicht beschreibt, der $score_i$ die Bewertung des Objektes bei der i-ten Wahrnehmung und $threshold_i$ der damals aktuelle Schwellenwert im Kurzzeitgedächtnis. Die Signifikanz soll ein Mass bilden wie wichtig ein Objekt für eine Ansicht ist.

Die in der Ansicht gespeicherten Masse eines gesehenen Objektes, wie Signifikanz und ungefährte Distanz zum Beobachter, werden gemittelt. Voraussetzung ist, dass das Objekt in aktuellen, sowie der gespeicherten Ansicht vorhanden ist. Wird das Objekt neu zugefügt, werden die neuen Masse übernommen. Ist das Objekt in der neuen Ansicht nicht mehr vorhanden, bleiben die Masse in der gespeicherten Version unverändert.

2.2.3 Der Schwellenwert

Ein Objekt, das sich im Kurzzeitgedächtnis befindet, kann vom Individuum erst wahrgenommen werden, wenn die Bewertung einen bestimmten Schwellenwert übersteigt. Dieser Schwellenwert sei adaptiv (Figure 2.11. Die nötige Mindestbewertung soll von der lokalen Umgebung abhängig sein. Diese Umfasst alle im Kurzzeitgedächtnis vorhandenen Objekte. Der Schwellenwert ist als die durchschnittliche Bewertung der Objekte dort gegeben. Er wird sich somit den aktuellen Wahrnehmungen anpassen.
Dies ermöglicht, dass auch in einer sehr kargen Umgebung Objekte noch wahrgenommen werden. In einer üppigen Umgebung tragen jedoch nur die herausragendsten Objekte zur individuellen Wahrnehmung bei. Objekte mit einer Bewertung über dem Schwellenwert drücken diesen nach oben, während Objekte unter dem Schwellenwert diesen nach unten ziehen. Der Schwellenwert ist jeweils an eine gewissen Adaptionszeit gebunden.
Falls über die Zeit die Bewertung der Objekte stark ansteigt, ist das Individuum besonders empfindlich, da der Schwellenwert erst mit einer zeitlichen Verzögerung mitzieht. Umgekehrt reagiert ein Individuum eher schwach auf seine Umwelt, falls der Bewertung der Objekte über die Zeit stark abfällt. Objekte mit kleiner Bewertung werden erst wieder erkannt, wenn der Schwellenwert sich nach unten angepasst hat.
Wir nehmen den Schwellenwert als arithmetisches Mittel aller Bewertungen von den im Kurzzeitgedächtnis gespeicherten Objekten an:

$$threshold = \frac{\sum_{i=1}^{n} score_i}{n}, \tag{2.2}$$

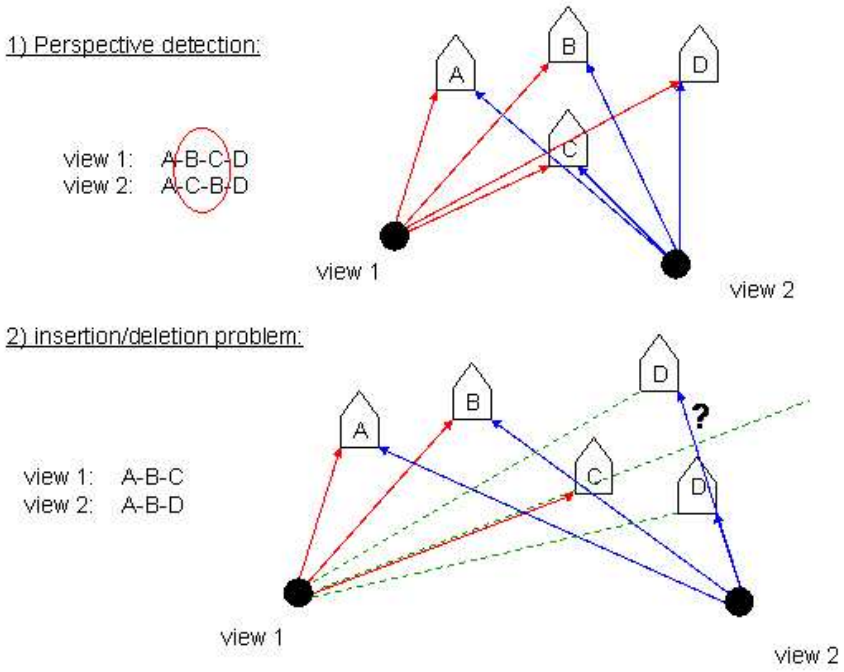


Abbildung 2.10: Der erste Teil der Grafik zeigt zwei Ansichten einer vereinfachten Szene mit Häusern, die sich in der Perspektive unterscheiden. Perspektiven sind definiert als Objektfolgen bei denen die Reihenfolge von mindestens einem Objektpaar vertauscht ist (roter Kreis). Der Zweite Teil der Grafik veranschaulicht das Problem, wenn Objekte in der Reihenfolge aufeinanderfolgend gelöscht und eingefügt werden. Man sieht deutlich, dass die topologische Ordnung verloren geht. Die betroffenen Objekte derjenigen Ansicht werden übernommen, die eine grössere Signifikant haben, im Kontext ihrer Ansicht.

wobei $threshold$ den Schwellenwert angibt, der Parameter n der Anzahl Elemente im Kurzzeitgedächtnis entspricht und $score_i$ der Bewertung des i'ten Objektes. Der Schwellenwert wird beim Hinzufügen eines neuen Elements jeweils angepasst.

2.2.4 Kurzzeitgedächtnis oder STM (short term memory)

Damit ein Objekt vom Agenten bewusst wahrgenommen werden kann, muss es in Kurzzeitgedächtnis eine bestimmte Bewertung übersteigen. Diese ist von verschiedenen Faktoren abhängig (Abbildung 2.12):

- Dauer der Wahrnehmung (Wie stark adaptiert das Objekt mit dem Hintergrund)
- vergangene Zeit seit der letzten Wahrnehmung ('power law of forgetting')
- Winkel zur Blickrichtung
- Sichtbarkeit (relative Fläche zur gesamtem Szene)
- Präferenz des Individuums für einen Objekttyp
- Entscheidungsdruck eines Individuums

Diese Faktoren sollen einen Selektionsmechanismus definieren, der die wichtigsten funktionalen Aspekte der natürlichen Informationsselektion wiedergibt. Der natürliche Berechnungsvorgang im Gehirn eines Individuums basiert auf einer hoch parallelisierten und hierarchisierten Netzwerkstruktur und ist deshalb sehr schnell. Der Simulation steht nichts dergleichen zu Verfügung. Eine funktionale Ann"herung muss genügen.

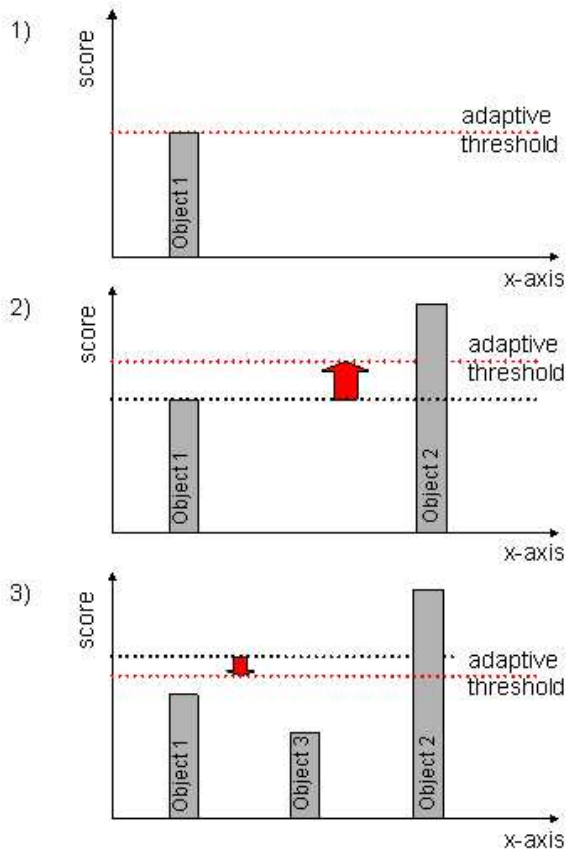


Abbildung 2.11: Der este Teil der Grafik zeigt den Schwellenwert gleich wie die Bewertung des ersten wahrgenommenen Objektes. Im zweiten Teil der Grafik wird zusätzlich zum ersten Objekt ein zweites von grßserer Bewertung wahrgenommen. Der Schwellenwert wird nach oben gedrückt vom neuen Objekt. Der dritte Teil der Grafik beinhaltet ein weiteres wahrgenommenes Objekt mit kleiner Bewertung. Der Schwellenwert wird wieder nach unten gezogen. Ein adaptiver Schwellenwert kann Probleme verursachen bei Szenen mit sehr wenig wahrgenommenen Objekten. Wichtige Information werden aufgrund des adaptiven Schwellenwertes weggelassen. Eine Adaption bringt auch eine gewisse Adaptionsphase mit sich, wo die Wahrnehmung, entweder überempfindlich ist oder kaum reagiert. Ersteres ist der Fall, wenn der Bewertungszuwachs der Objekte pltzlich viel stärker wird, letzteres, falls der Bewertungszuwachs auf einmal sehr klein wird.

Jedes Mal, wenn der Agent ein bestimmtes Objekt wahrnimmt, wird dessen Bewertung im Kurzzeitgedächtnis aktualisiert. Dabei geht man von einer anfänglichen Bewertung von null aus. Die Bewertung von $Objekte_i$ wird nach jeder Wahrnehmung um den Betrag $\Delta score_i$ erhöht. Die Gleichung 2.3 gibt die Details, wie aus den besprochenen wahrnehmungsspezifischen Faktoren der Bewertungszuwachs berechnet wird.

$$\Delta score_i = (1 - \frac{|angle|}{180}) * preference_i * visibility * decisionPressure * adaptation, \tag{2.3}$$

wobei $\Delta score_i$ den Bewertungszuwachs von $Objekt_i$ im STM ist, $angle$ der Winkel ($\in [-180, 180]$)relativ zur Blickrichtung des Agenten, $preference$ die persönliche Präferenz für den Objekttyp, $visibility$ die Sichtbarkeit relativ zum gesamten Gesichtsfeld

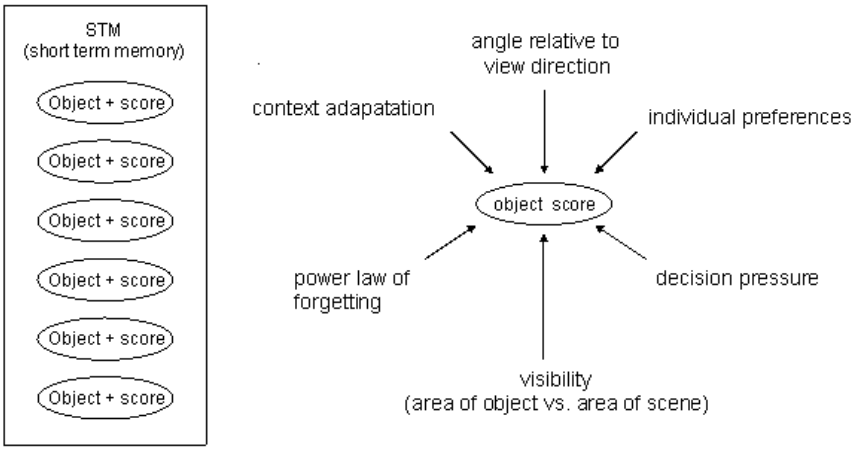


Abbildung 2.12: Das Kurzzeitgedächtnis (STM) ist eine Ansammlung von Objekten mit einer Bewertung (='score') (linke Seite der Grafik). Diese Bewertung muss einen bestimmten Schwellenwert überschreiten, damit das Objekt auch bewusst wahrgenommen wird. Dieser Schwellenwert ist adaptiv. Er wird von der Bewertung der anderen Objekte im STM beeinflusst. Der Bewertung selbst ist von verschiedenen Faktoren abhängig (links)

($\in [0,1]$), $decisionPressure$ der Entscheidungsdruck ($\in \mathbb{R} > 0$) und $adaptation$ der Adaptionsfaktor. Letzterer gibt an, wie stark $Objekt_i$ dem Hintergrund zugewiesen werden kann.

Je häufiger über eine vorgegebene Zeitspanne das Objekt wahrgenommen wird, desto schneller steigt die Bewertung. Nimmt der Agent ein Objekt über eine längere Zeitspanne mit hoher Frequenz war, verliert dieses jedoch an Signifikanz für die momentane Ansicht. Mit anderen Worten: je stärker ein Objekt dem lokalen Hintergrund zugewiesen werden kann, umso weniger ist es charakteristisch für eine bestimmte Ansicht. In der Simulation wird die Hintergrundadaption in der dazugewonnenen Bewertung bei der Wahrnehmung eines Objektes umgesetzt. Dazu wird der Bewertungszuwachs mit einem Faktor $adaption \in [0,1]$ gewichtet (vgl. Gleichung 2.3). Dieser berechnet sich folgendermassen:

$$adaptation = -\frac{1}{v_{adaptation} * \Delta t + 1} + 1, \tag{2.4}$$

wobei $v_{adaptation}$ der konstanten Adaptionsgeschwindigkeit entspricht und Δt das Zeitintervall zwischen dieser und der letzen Wahrnehmung des Objekts durch den Agenten wiedergibt. Je höher die Frequenz mit der ein Objekt wahrgenommen wird, desto mehr konvergiert die Gewichtung ('adaptation') des Bewertungszuwachs gegen null und umgekehrt gegen eins.

Damit ein $Objekt_i$ nicht permanent im Kurzzeitgedchtnis bleibt, soll dessen Bewertung $score_i$ auch die Möglichkeit haben, wieder abzunehmen. Um dies zu erreichen, wird die Bewertung von einer von der Zeit abhängigen Zerfallsfunktion abhängig gemacht. Man nennt diesen Vorgang auch 'vergessen'. Das in Sektion 1.3 besprochene 'power law of forgetting' wird angewendet. Mit der folgenden Formel wird ein exponentiell abfallender Zerfallsfaktor $\in [0,1]$ für die Bewertung eines Objektes angenähert.

$$decay_i = \frac{sensitivity + (t_{i1} - t_{i0})}{sensitivity + (t_{i2} - t_{i0})}, \tag{2.5}$$

wobei $decay_i$ den Zerfallsfaktor der Bewertung zum Zeitpunkt t_{i2} über das Zeitintervall $t_{i1} - t_{i2}$ angibt. t_{i0} ist der Zeitpunkt an dem das Objekt zuletzt wahrgenommen wurde. Bei einer erneuten Wahrnehmung eines Objektes wird jeweils auch dessen Zerfallsfunktion neu initialisiert. Die Zeiten t_{i2} und t_{i1} werden relativ zu t_{i0} gemessen. Der Parameter $sensitivity$ steht für die Sensitivität der Funktion und charakterisiert die Zerfallsgeschwindigkeit einer Information.

Der Zerfallsfaktor wird auf die Bewertung der Information angewandt. Dies geschieht nur dann, falls die Informationsbewertung abgefragt wird, oder bevor die Bewertung aktualisiert wird. Letzteres ist der Fall, falls ein Objekt erneut wahrgenommen wird. Durch diese Form von 'lazy evaluation' soll Rechenzeit gespart werden. Ansonsten wäre eine regelmässige Berechnung des Zerfalls nötig. Dabei müsste jeweils über die gesamten Einträge iteriert werden.
Durch den begrenzten Speicherplatz muss das Kurzzeitgedchnis von Zeit zu Zeit 'aufgeräumt' werden. Dies kann durch verschiedene Strategien erreicht werden. Die offensichtlichste ist, den jeweils schwächsten Eintrag zu überschreiben, sobald das Speicherlimit erreicht wird. In der Simulation wird eine andere, ähnliche Strategie verwendet. Sobald das Limit erreicht ist, werden alle Einträge unterhalb des Schwellenwertes gelöscht. Dies hat einen reduzierten Suchaufwand zur Folge. Ansonsten müsste, ist einmal das Limit erreicht, bei jedem neuen Eintrag wieder derjenige mit der schlechtesten Bewertung gesucht und anschliessend gelöscht werden. Die Natur überragt hier mit ihrem hohen Parallelisierungsgrad der vernetzten Neuronen, was den Suchvorgang enorm beschleunigt.
Die wahrgenommenen Objekte können also nur durch repetive Wahrnehmung im Kurzzeitgedächtnis gehalten werden. Mit anderen Worten, ein Objekt, dass regelmässig vom Agenten gesehen wird, erhält jedes mal ein Bewertungszuwachs im Kurzzeitgedächtnis. Die Bewertung kann so nie richtig zerfallen. Wird das Kurzzeitgedächtnis aufgeräumt, ist die Chance gross, dass die Bewertung über dem erforderlichen Limit liegt.

2.3 Hierarchische Strukturierung der Information

Die Simulation verwendet Hierarchisierung nur in beschränktem Mass. Oft ist es nicht trivial, die einzelnen Hierarchiestufen zu formulieren und zu erkennen. Ein Beispiel wäre ein Objekt 'Berg'. In der Speicherstruktur der Simulation wird eine einfache Aggregationshierarchie verwendet. Wahrgenommene Objekte werden zu Ansichten aggregiert, welche wiederum 'action links' definieren, woraus Routenobjekte generiert werden können.
Weiter wird eine Generalisierungshierarchie für die Typinformationen verwendet. Die Hierarchie assoziiert durch Generalisation die Typinformationen sämtlicher möglichen Objekttypen, die wahrgenommen werden können. Das Generalisierungswissen ist fest in der Simulation verankert und wird mit dem Initialisierungsvorgang aktiviert. Die Hierarchie unterstützt Mehrfachvererbung. Nicht nur Typinformationen, sondern auch Tätigkeiten, z.B. Aktivitäten, können so assoziiert werden.
Jedes vom Agenten wahrgenommene Objekt besitzt einen bestimmten Typ. Mit Hilfe des Generalisationswissens können Objekte mit bestimmtem Supertyp oder die für eine bestimmte Aktivität geeignet sind aggregiert werden. Abbildung 2.13 zeigt ein Beispiel einer Hierarchiedefinition, wie sie in der Simulation verwendet wird.
Im folgenden Beispiel soll die Anwendung des Generalisierungswissens deutlich werden. Wenn der Agent z.B. ein Restaurant besuchen will, so kann dies relativ einfach bewerkstelligt werden, indem er nach Objekten mit dem Supertyp 'Restaurant' sucht. Es ist dann dem Agenten überlassen, in welches er schlussendlich gehen will. Ebenso kann uns die Generalisierungshierarchie auch mitteilen, ob es sich hier auch um eine Essensgelegenheit handelt, falls ein Supertyp 'essen' mit dem Typ 'Restaurant' assoziiert ist.
Ein Individuum leitet Informationen nicht nur aus der direkten Wahrnehmung von Objekten ab, sondern lernt solches Grundwissen über verschiedenste Umwege, wie z.B. Kommuni-

```
<world>
    <still>
        <stillactivities>
            <eat>
                <restaurant/>
            </eat>
        </stillactivities>
        <object>
            <traversable>
                <building>
                    <church/>
                    <school/>
                    <restaurant/>
                    <hotel/>
                </building>
                <route>
                    <intersection/>
                    <street/>
                </route>
            </traversable>
            <nontraversable>
                <tree/>
                <sign/>
            </nontraversable>
        </object>
    </still>
    <move>
        <moveactivities>
            <hike/>
        </moveactivities>
    </move>
</world>
```

Abbildung 2.13: Die Abbildung zeigt eine Gereralisationshierarchie in XML Format. Verschiedene Objekttypen, die der Agent wahrnehmen kann, sind definiert. Dies sind z.B. ein Restaurant, eine Strasse, ein Hotel, eine Kreuzung. Generalisationstypen können einerseits Verallgemeinerungen von Objekten sein oder Aktivitäten, die mit diesen Objekten assoziiert sind. Der Typ 'Restaurant' ist in der Abbildung einerseits als eine Essensgelegenheit definiert, da es 'eat' als Supertyp hat, anderseits aber auch als ein Gebäude ('building').

kation oder via externe Quellen wie 'Hotelführer' etc.. Solche Wahrnehmungsfähigkeiten sind in der Simulation nicht vorhanden. Die Verwendung von vorgegebenem Wissen ist deshalb gerechtfertigt.
Die Möglichkeiten der Hierarchisierung von Informationen wird in der Simulation nur in bescheidenem Masse ausgeschöpft. Man könnte sich vorstellen, dass der Agent sich auf verschiedenen Hierarchiestufen orientieren kann. Dies würde ihm erlauben, Operationen auf einer höheren Abstraktionsebene zu betrachten und nur im Detail auszuführen, wo die entsprechenden Informationen in groberer Form vorhanden sind (vgl. Sektion 1.5). Die mentale Karte müsste dann das räumliche Wissen auf mehreren Hierarchiestufen erkennen und speichern.

2.4 Bewertung von Informationen - Attraktivität

Die Bewertung von wahrgenommenen Informationen mit Einbezug persönlicher Präferenzen wird als Attraktivität definiert. Die Attraktivität ist eine individuelle Empfindung. Der Agent in der Simulation bewertet Basisinformationen im Kurzzeitgedächtnis, sowie Informationen auf abstrakterer Ebene, die das emotionale Empfinden ansprechen, wie z.B. 'schöne Aussicht'.
Basisinformationen sind definiert als die von der Simulation direkt wahrgenommenen Objektansichten, die 'view'-events (vgl. Abbildung 2.2). Die Attraktivität ist dort einerseits von der Frequenz, mit der das Objekt wahrgenommen wird, von räumlich metrischen Informationen und individuellen Präferenzen des Agenten für ein Objekttyp abhängig (Gleichung 2.3). Die durchschnittliche Bewertung der im Kurzzeitgedächtnis gespeicherten

Objekte entscheidet, ob dem Agenten das wahrgenommene Objekt bewusst wird. Dieser Vorgang der Informationsselektion wird in Untersektion 2.2 beschrieben.
Wie bereits erwähnt, restriktieren sich die vom Agenten wahrgenommenen Informationen nicht nur auf die Basisinformation, sondern auch auf Informationen, die das emotionale Empfinden ansprechen. Solche emotionalen Informationen werden von der Simulation berechnet, die 'feel'-events (vgl. Abbildung 2.2). Die Attraktivität einer solchen Information ist wiederum individuell vom Agenten abhängig. Die Simulation liefert nur die Daten, wann und wo eine solche Empfindung stattgefunden hat. Der Agent entscheidet aufgrund der persönlichen Präferenzen und der Frequenz, mit der er den Empfindungstyp wahrnimmt, deren Stärke. Die Empfindung wird mit der räumlichen Position des Agenten assoziiert. Die Attraktivitäten werden für die Aktivitätenplanung wiederverwendet. In Anbetracht der Aufgabe ist es sinnvoll, die einzelnen Empfindungen den verschiedenen Routenobjekten in der mentalen Karte zuzuweisen
Die Attraktivität einer Information ist unabhängig oder konstant, wenn sie nicht von anderen bewerteten Informationen beeinflusst wird und selber in einem regelmässigen Muster mit räumlicher und zeitlicher Dimension erscheint. Mit anderen Worten, jedesmal wenn der Agent eine bestimmte Information entweder an einem bestimmten Ort zu einer bestimmten Zeit wahrnimmt oder die schon gespeicherte Information aus der mentalen Karte abfragt, muss deren Bewertung gleich sein.
Solche unabhängigen oder konstanten Attraktivitäten stehen im Widerspruch zu der 'Aufgabenbedingten Wahrnehmung', wie sie in Sektion 1.3 besprochen wurde. Demnach ist die Selektion von Informationen aus einer visuellen Szene oder aus der mentalen Karte auch von der momentanen 'Aufgabe im Kopf' abhänging. Die Aufgabe im Kopf ist wiederum abhängig vom Empfindungszustand des Agenten und damit von Informationsbewertungen, d.h. Attraktivtäten, die diesen Zustand beeinflussen. Mit anderen Worten kann also eine spontane Änderung einer Attraktivität, den Empfindungszustand so beeinflussen, dass der Agent seine Aufgabe ändert. Wenn wir davon ausgehen, dass sich die Aktivitäten zusätzlich gegenseitig beeinflussen, wird dieser Effekt noch verstärkt.
Folgendes Beispiel verdeutlicht den Sachverhalt: Falls die Sonne scheint, wird ein Wanderer die schöne Aussicht geniessen. Er hat sich deshalb zur Aufgabe gemacht, möglichst viel von dieser schönen Aussicht auf seiner Wanderung zu konsumieren. Wird die Sonne durch aufziehende Wolken verdeckt, beeinflusst dies den Empfindungszustand des Agenten. Der Agent nimmt nun entgegen der ursprünglichen Erwartung nur noch wenig oder keine Sonne mehr wahr. Zusätzlich bewertet er die Sonnenabhängigen Attraktivitäten ebenfalls schlechter, was den Einfluss auf den Empfindungszustand noch verstärkt. Der Empfindungszustand wiederspiegelt sich dabei in der Bewertung der einzelnen Routenobjekte. Sie entsteht durch die Assoziation der einzelnen räumlich darauf wahrgenommenen Attraktivitäten durch den Agenten.
Konkret auf die Simulation angewandt bedeutet dies: Verändert sich der Empfindungszustand des Agenten, so ändert sich auch Bewertung der einzelnen Routenobjekte in der mentalen Karte. Dies beeinflusst die Wahl der Route und damit auch der entsprechenden Objekte in der mentalen Karte zur Ausübung bestimmter Aktivitäten. Mit der Maximierung der neuen Bewertung wird die Route aktualisiert. Das veränderte Verhalten des Agenten kommt einer 'Änderung der Aufgabe im Kopf' gleich. In der Simulation ist der Eindruck der neuen Aufgabe etwas eingeschränkt, da der Agent gezwungen wird, sich möglichst an vorgegeben Aktivitäten zu halten. Es wäre durchaus vorstellbar, die Aktivitäten auf gleiche Weise abhängig zu gestalten.

2.4.1 Statische Muster - unabhängige Attraktivitäten

Unabhängige und konstante Attraktivitätsmuster von Informationen haben neben der räumlichen Komponente, die gegeben durch die Datenstruktur ist, auch eine zeitliche Komponente. Sie folgen einem regelmässigen Muster. Die Attraktivitäten werden auf ein Zeitfenster von einem Tag aufgeteilt, wobei die Skalierung auf Stunden festgelegt ist (vgl. Sek-

tion 1.7). Die maximale Attraktiviät ist auf eins normiert. Die Grösse der Attraktivität ist durch die Häufigkeit des entsprechenden Empfindungsevents festgelegt. Diese wird dabei abgeschätzt als die Anzahl Events einer Attraktivität i, die während der aktuellen Stunde detektiert wurden. Werden Stunden an denen noch nie ein Empfindungsevent empfangen wurde abgefragt, geben wir den Durchschnitt der bisher gemessenen Häufigkeiten zurück. Die folgende Gleichung 2.6 wird zur Berechnung der Attraktivität verwendet:

$$attractivity_i^t = (1.0 - \frac{1.0}{\sum_{t=start_{hour}}^{end_{hour}} events_{attractivity_i^t}}) * preference_i, \qquad (2.6)$$

wobei $attractivity_i^t$ die Attraktivität eines Routenobjektes zu einer bestimmten Stunde t angibt, $events_{attractivity_i^t}$ die Anzahl des Empfindungs-Events i ('feel events') in der entsprechenden Stunde und $preference_i$ die Präferenz für die Attraktivität i. Dies gilt nur für Zeiten an denen bereits ein Empfindungsevent wahrgenommen wurde. Wurde bei einer Stunde für einen Attraktivitätstypen kein Empfindungsevent detektiert, wird für die Häufigkeit in der Attraktivitätsberechung des Empfindungsevents der Durchschnitt der Stunden genommen, wo bereits entsprechende Events detektiert wurden.

$Attractivity_i$ in Gleichung 2.7 ist die Attraktivität mit vernachlässigter Zeitdimension. Die Attraktivität mit zeitlicher Abhängigkeit verlangt viel mehr Daten, um gut angenähert zu werden. Anderseits muss die Simulation die Attraktivitäten nach einem zeitabhängigen Muster generieren. Beides ist momentan nicht verfügbar. In der Simulation bedeutet dies, dass die Häufigkeit an den einzelnen Tagestunden jeweils gleich sein wird, da immer gleich viel Empfindungsevents an einem Ort wahrgenommen werden. Die Durchschnittshäufigkeit, die bei Stunden, an denen noch keine Empfingungsevents detektiert wurden, verwendet wird, ist somit auch identisch mit den anderen Häufigkeiten. Im Weiteren vernachlässigen wir deshalb die zeitabhängige Schreibweise (vgl. Gleichung 2.7).

$$attractivity_i = (1.0 - \frac{1.0}{hours_{used} \cdot events_{attractivity_i}}) * preference_i, \qquad (2.7)$$

wobei $attractivity_i$ die Attraktivität eines Routenobjektes angibt, $hours_{used}$ die Anzahl angeschnittenen Stunden während die Route abgelaufen wurde, $events_{attractivity_i}$ die Anzahl des Empfindungs-Events i ('feel events'), die auf der Route wahrgenommen werden und an jedem Zeitpunkt gleich ist sowie $preference_i$ die Präferenz für die Attraktivität i.

Die Attraktivität eines Routenobjektes setzt sich zusammen aus verschiedenen Teilattraktivitäten. Für jeden Empfindungstyp wird ein Attraktivitätsmuster angelegt. Eine Strecke einer Route ist z.B. charakterisiert durch die Attraktivität für 'schöner Wald', 'schöne Aussicht' und andere. Der Durchschnitt der einzelnen Attraktivitätstypen ergibt die Objektattraktivität. Die Aufgabe des Agenten und der zugehöhrige Lösungsalgorithmus bestimmen, ob die Gesammtattraktivität oder nur Teilattraktivitäten verwendet werden.
Wird ein Routenobjekt mehrmals wahrgenommen, werden die Attraktivitäten des aktuellen Routenobjektes jeweils gemittelt mit denjenigen der gespeicherten Version.

2.4.2 Dynamische Muster - gegenseitig abhängige Attraktivitäten

In Sektion 2.4 werden dynamisch veränderbare Bewertungen angesprochen. Diese können sich gegenseitig beeinflussen. Ein Beispiel war die 'schöne Aussicht' assoziiert mit 'Sonnenschein'. Es besteht eine gegenseitige Abhängigkeit. Wird die Sonne durch aufziehede Wolken verdeckt, verändert sich die Attraktivität für 'Sonnenschein'. Je nach dem, wie

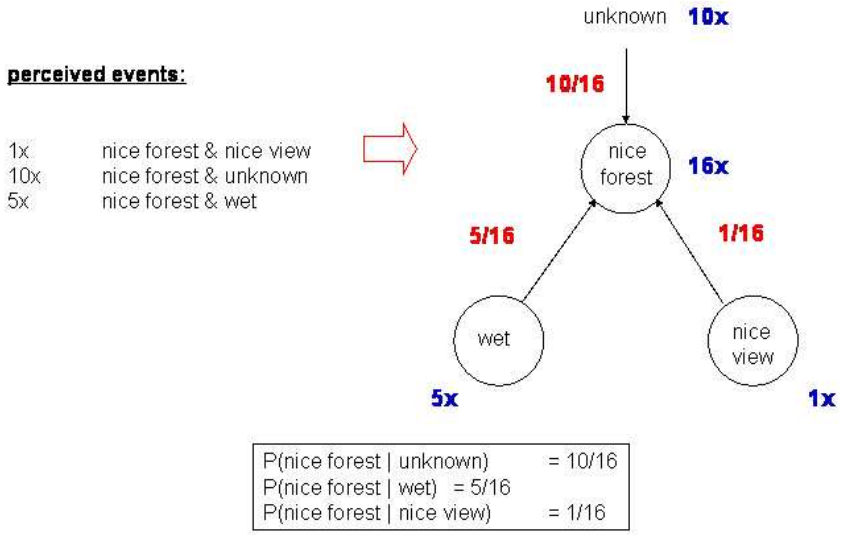


Abbildung 2.14: Falls der Agent ein Empfindungs-Events empfängt, während ein anderer Empfindungstyp aktiv ist, so besteht eine Abhängigkeit. Je nach dem, wie oft dies eintrifft, umso stärker wird die Abhängigkeit. Wird ein Empfindungsevent empfangen, ohne dass ein anderes Empfindungsevent aktiv ist, so besteht eine Verbindung zu *Unbekannt*. Unbekannt heisst irgendeine Abhängigkeit zu einen Attraktivitätstyp der in der Simulation nicht definiert ist. Es existiert also bei jeder Wahrnehmung mindestens eine Abhängigkeit. Die Summe aller Abbhängigkeiten von einer bestimmten Attraktivität ist eins. Die Abhängigkeit zwischen dem Empfindungstyp A und B ist als Wahrscheinlichkeiten P(A,B) modelliert. Empfängt man den Empfindungstyp B, so ist die bedingte Wahrscheinlichkeit, P(A|B), die Wahrscheinlichkeit, dass der Empfindungstyp A aktiv ist. Diese Wahrscheinlichkeit wird angenähert durch das Verhältniss von Empfindungen A wahrgenommen, während die andere Empfindung B aktiv ist, zu der Anzahl Wahrnehmungen der Empfindung A.

stark die 'schöne Aussicht' von 'Sonnenschein' abhängig ist, ändert auch diese Attraktivität.
Folgen diese dynamischen Veränderungen von Empfindungsinformationen selbst einem bestimmten räumlichen und zeitlichen Muster, tun es auch die einzelnen Attraktivitäten in der Simulationswelt betrachtet man wiederum die räumliche und zeitliche Dimension.
Es gibt verschiedene Ansätze, das Lernen von gengenseitigen Abhängigkeiten von Informationen modellieren. Beispiele sind z.B. 'Markov Models','Hidden Markov Models' (Rabiner [11]) oder Bays'sche Netzwerke (Jensen [6]). In der Simulation wird ein Ansatz, der eine Kombination der beiden ist, verwendet. Es lassen sich die Grundprinzipien beider Ansätze wiederfinden. Ziel ist, die Abhängigkeiten der verschiedenen Attraktivitäten zu modellieren. Es soll generell bekannt sein, wie stark z.B. die Information 'Sonnenschein' durchschnittlich die 'schöne Aussicht' beeinflusst. Der Agent kann, wenn er merkt, dass er entgegen der Erwartung in der mentalen Karte weniger Sonne empfindet, darauf schliessen, dass höchstwahrscheinlich auch keine so schöne Aussicht mehr wahrgenommen wird. Durch die daraus resultierende Neubewertung der Informationen, die sich aus der verfehlten Erwartung für 'Sonneschein' und der davon abhängigen Attraktivitäten ergibt, werden nun andere Informationen für die Aktivitätenplanung bevorzugt.
Eine Attraktivität sei beeinflussbar von anderen bekannten Attraktivitäten und von einer unbekannten Seite. Es entsteht ein gerichteter Graph, der den Informationseinfluss zu der Attraktivität darstellt. Dieser Informationseinfluss zwischen einer Attraktivität A und einer Attraktivität B, wird durch die Wahrscheinlichkeit P(A,B) dargestellt. Sind alle Attraktivitäten B die Einfluss haben aktiv, so sollte der Agent für die gespeicherten Informationen die maximale Attraktivität wahrnehmen. Wird eine Attraktivität wahrgenommen, so bleibt

sie über einen kurzen Zeitraum aktiv.

Anmerkung: Im Folgenden wird die zeitliche Dimension der Attraktivität vernachlässigt.

Es gilt dabei aus Bayes'Netzwerke Theorie:

$$P(attractivity_i) = \sum_{j=1}^{n} P(attractivity_i, attractivity_j) \quad i \neq j, \tag{2.8}$$

wobei $P(attractivity_i)$ die Wahrscheinlichkeit ist die Attraktivität i wahrzunehmen und $P(attractivity_i, attractivity_j)$ die Wahrscheinlichkeit, dass Attraktivtät i wahrgenommen wird und Attraktivität j noch aktiv ist.

$$P(attractivity_i) = \sum_{j=1}^{n} P(attractivity_i | attractivity_j) \cdot P(attractivity_j) \quad i \neq j, \tag{2.9}$$

wobei $P(attractivity_i)$ die Wahrscheinlichkeit die Attraktivität i wahrzunehmen ist und gelten soll $P(attractivity_i) = 1$ falls alle $P(attractivity_j) = 1$

Falls in unserem Modell die Wahrscheinlichkeit für eine Attraktivität nur beeinflusst wird von den Attraktivitäten, die direkt abhängig sind von dieser, so betrachten wir ein Markov Modell ersten Grades (Abbildung 2.14). Die Verwendung von solch einfachsten Netzwerken ist von Vorteil, da sich der Umgang mit Bayes'Netzwerken schnell verkompliziert, sobald das Informationsnetzwerk keine Baumstruktur hat (Jensen [6]).
Legt man die verschiedenen Teilgraphen für die einzelnen Attraktivitäten übereinander, so entsteht ein vollständiger Graph, der die einzelnen Abhängigkeiten darstellt (Abbildung 2.15). Daraus lässt sich eine Übergangsmatrix formulieren mit den Üebergangswahrscheinlichkeiten zwischen den einzelnen Zuständen (Einführungskapitel in Rabiner [11]).
Es wird angenommen, dass eine Attraktivität nur in Abhängigkeit einer anderen Attraktivität vorkommen kann, im Zweifelsfall sei dies eine unbekannte Attraktivität. Wird eine Aktivität eines bestimmten Typs wahrgenommen, bleibt dieser über eine bestimmte Zeit aktiv. Der Aktivitätstyp hat eine Bewertung die sich mit jeder Wahrnehmung einer entsprechenden Aktivität um einen konstanten Betrag vergrössert und über die Zeit wieder zerfällt (vgl. Prinzip mit Kurzzeitgedächtnis in Sektion 2.2.4). Ein Attraktivitätstyp ist aktiv falls die Bewertung über einem konstanten vorgegebenen Schwellenwertliegt. Ist bei einem wahrgenommenen Attraktivitätsereignis kein Attraktivitätstyp aktiv, so wird eine Abhängigkeit zu einer 'unbekannten' Attraktivität vermutet.
In der Simulation werden nicht direkt die Wahrscheinlichkeiten in der Übergangsmatix gespeichert. Der ij-te Eintrag in der Matrix hält die Anzahl, mit der die $Attraktivität_i$ und die $Attraktivität_j$ zusammen wahrgenommen wurden. Der Eintrag mit dem Index auf sich selber wäre leer, da eine Attraktivität nicht von sich selbst abhängig sein kann. Er freie Eintrag wird mit der Abhängigkeit zu Unbekannt aufgefüllt (in den Gleichungen 2.8, 2.9, 2.11, 2.11 wird Unbekannt jedoch als separater Index betrachtet).
Solange der Agent aktiv ist, wird sich die Übergangsmatrix ständig ändern. Man spricht auch von 'online learning'. Die Übergangswahrscheinlichkeiten werden dabei folgendermassen berechnet:

$$P(attractivity_i | attractivity_j) = \frac{\sum_{time=0}^{now} attractivity_{ij}^{time}}{\sum_{time=0}^{now} attractivity_i^{time}}, \tag{2.10}$$

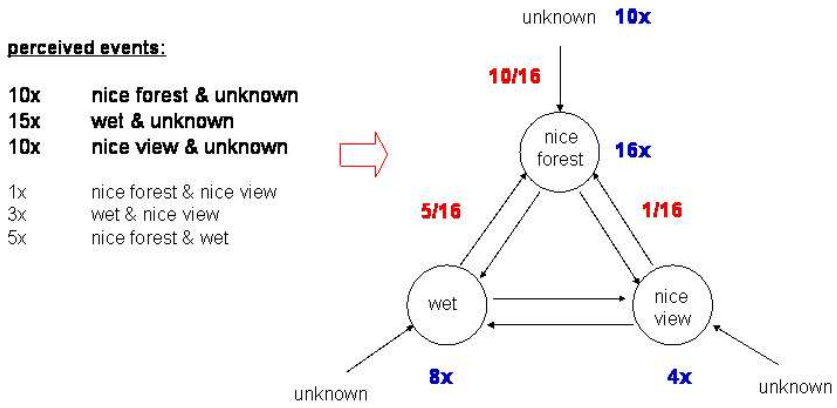


Abbildung 2.15: Durch die Überlagerungen der einzelnen Beziehungsmuster entsteht ein Netzwerk, d.h. ein vollständger gerichteter Graph, der die Beziehungen der einzelnen Attraktivitäten wiedergibt. Es kann eine Übergangsmatrix der Attraktivitäten formuliert werden. Dabei werden die Beziehungen zu 'Unbekannt' jeweils als Verbindung auf sich selbst dargestellt.

wobei $P(attractivity_i|attractivity_j)$ die Wahrscheinlichkeit ist die Attraktivität i wahrzunehmen falls Attraktivität j wahrgenommen wird. $Attractivity_{ij}$ ist die Anzahl der Wahrnehmungen von $Attrakivität_i$ zusammen mit $Attraktivität_j$ und in der Matrix in dem ij-ten Eintrag wiedergegeben. Der ij-te Eintrag in der Matrix wird um eins erhöht, falls eine $Attraktivität_i$ zu einem bestimmten Zeitpunkt $time$ vom Agenten wahrgenommen wird und die $Attraktivität_j$ noch aktiv ist, sonst null.

Die emotionale Attraktivität einer Information wird dann folgendermassen berechnet:

$$\begin{aligned} & attractivity_i \\ &= score_i \cdot \textstyle\sum_{j=1}^{n} P(attractivity_i, attractivity_j) \\ &= score_i \cdot \textstyle\sum_{j=1}^{n} P(attractivity_i|attractivity_j) \cdot P(attractivity_j) \quad i \neq j, \end{aligned} \tag{2.11}$$

wobei $P(attractivity_j)$ gleich eins ist, falls sie aktiv ist im Kurzzeitgedächtnis, sonst null, und $score_i$ sei die Bewertung der unabhängigen Attraktivität (vgl. Gleichung 2.6). Die unbekannte Attraktivität sei per Definition immer aktiv bei der Attraktivitätsberechnung.

Beinhaltet das System Informationen, die Bewertungen beeinflussen und nicht einem regelmässigen Muster folgen, so beeinflusst diese Unregelmässigkeit Bewertungen, die eigentlich einem regelmässigen Muster folgen. Diese verlieren so ihre Regelmässigkeit.

2.4.3 Zeitabhängige Routenzeiten

In der Simulation wird für jedes Routenobjekt eine bestimmte Zeit gespeichert. Die Zeit kann theoretisch Varieren, je nach dem zu welcher Tageszeit der Agent das Routenobjekt abläuft. Es ist vorstellbar, dass es Tageszeiten gibt, wo es z.B. mehr Agenten hat. Durch den entstehenden Stau wird dann länger für eine Route gebraucht. Es deshalb sinnvoll die Zeit , die für eine Route benötigt wird, anhängig von der Tageszeit zu speichern. In der Simulation wird dabei, wie schon in Sektion 2.4.1, die Auflösung auf Stunden festgelegt. Die Zeiten für ein Routenobjekt werden also bezüglich der Startstunde abgelegt. Wird die Zeit für ein Routenobjekt abgefragt, welches noch nie begangen wurde, nehmen wir den Durchschnitt, der bereits bekannten Zeiten.

Die Simulation wird nur mit wenigen Agenten getestet. Die Zeiten für das Ablaufen von Routenobjekten verlieren dadaruch ihre Zeitabhängigkeit. Die Zeitabhängigkeit wird im Weitern deshalb nicht mehr speziell erwähnt.

Kapitel 3

Wiederverwenden der Daten der mentalen Karte

Die Simulation soll mehrere Agenten koordinieren, die 'täglich' bestimmte Aktivitäten ausführen. Die Agenten verwenden für die Planung ihrer Tagesroute die in ihrer mentalen Karte gespeicherten räumlichen Informationen. Die Kriterien, die ein Agent verwendet, um auf seinem Wissen zu navigieren, sind deren individuelle Bewertung. Während des Planungsvorgangs werden die Bewertungen als konstant angenommen. (vgl. Sektion 2.4).

3.1 Extrahieren von Informationen aus einer individuellen Wissensbasis

Im Gegensatz zum Aquisieren von neuen Informationen, was Ansichtenbasiert ist, orientiert sich die Informationsextraktion auf der Stufe der einzelnen Objekte. Durch die spezielle Datenstruktur der mentalen Karte hat der Agent jeweils die lokale Umgebung eines beliebigen Objektes sehr schnell präsent.
Der Vorteil einer individuellen Wissensbasis ist, dass der Suchraum um eine Information zu finden viel keiner ist, als bei totalem Wissen über die Agentenwelt. Das Wissen ist jedoch unvollständig und bringt deshalb auch Limitationen in den Aktionsmöglichkeiten des Agenten mit sich. Was der Agent nicht kennt, kann er auch für seine Planung nicht verwenden. Dies entspricht dem Verhalten eines Individuums.
Mit der Vergrösserung dieses Wissens ändert sich das Verhalten des Agenten. Ein solcher Zugewinn von Informationen ist möglich, da der Agent Wissen in der Planung verwendet, das er direkt und indirekt erfahren hat. Das direkt erfahrene Wissen beinhaltet die Objekte, die der Agent effektiv besucht hat. Ein Beispiel für direkt erfahrenes Wissen ist z.B. das Restaurant, in dem der Agent gestern essen war. Im indirekten Wissen sind Objekte, die der Agent wahrgenommen, jedoch noch nie besucht hat, enthalten. Ein Beispiel wäre ein Restaurant, das der Agent gestern gesehen hat, als er auf der Strasse X wanderte. Die beiden Wissensarten unterscheiden sich in der Anzahl der Informationen, die für ein Objekt gesammelt wurden. Wir definieren diesen Unterschied als Unsicherheit über ein Objekt. Je weniger Informationen ein Agent über ein Objekt besitzt, desto grösser ist die Unsicherheit über das Objekt.
Der Agent plant basierend auf einem Aktivitätenplan. Ein Aktivitätenplan beinhaltet verschiedene Tätigkeiten, die der Agent an einem bestimmten Tag ausführen möchte. Mit dem ihm zur Verfügung stehenden Generalisationswissen versucht er die diese Aktivitäten bestimmten Objekten oder Routen aus der mentalen Karte zuzuweisen (vgl 2.3). Es müssen gewisse zeitliche Bedingungen für die Aktivitäten eingehalten werden.
Durch die Kommunikation mit anderen Agenten öffnet sich eine weitere Möglichkeit, das

persönliche Wissen zu erweitern. In der Simulation ist dies möglich, indem der Agent einen anderen Agenten beauftragt, seinen Aktivitätenplan mit mentalem Wissen zu komplettieren. Der Agent vertraut dabei blind seinem Kommunikationspartner.

```
<request type="plan" id="44">
    <plan>
        <act name="hotel" location_type="object_id" id="461" endtime="0" />
        <act name="hike" duration="3000"/>
        <act name="eat" duration="1800"/>
        <act name="hike" duration="3200"/>
        <act name="hotel" location_type="object_id" id="462" starttime="8000"/>
    </plan>
</request>
```

Abbildung 3.1: Ein Plan ist definiert als eine Reihenfolge von Aktivitäten. Der Agent generiert jeweils einen solchen Plan für jeden Tag. Er vervollständigt ihn mit den räumlichen und zeitlichen Informationen der mentalen Karte. Eine Planaktivität muss mindestens durch den Aktivitätentyp definiert sein. Es wird unterschieden zwischen punktuellen Aktivitäten und solchen die eine räumliche Ausdehnung haben. Der Startort eines Plans muss eindeutig definiert sein. Weiter muss auch eine gewünschte Zeit angegeben werden, die der Agent für eine bestimmte Aktivität aufbringen möchte.

3.2 Der Plan des Agenten

Die Aufgabe des Agenten ist, einen bestimmten Aktivitätenplan mit räumlichem Wissen aus der mentalen Karte zu vervollständigen. Eine Aktivität beschreibt dabei eine bestimmte Tätigkeit des Agenten. Sie ist definiert durch den Aktivitätentyp. Ein Beispiel für einen solchen Plan zeigt Abbildung 3.1. Im Anfrageplan muss mindestens die Startaktivität räumlich bestimmt sein. Genauso soll eine ungefähre erwünschte Dauer jeder Aktivtät angegeben werden. Folgende Bedingungen können oder müssen dem Agenten übergeben werden:

- Aktivitätentyp
- erwartete Dauer
- erwartete Bewertung
- Wegpunkt (nur bei punktuellen Aktivitäten)

Der Agent unterscheidet zwischen zwei Arten von Aktivitäten. Aktivitäten können einerseits sehr lokal, d.h. punktuell für einen bestimmten Ort bestimmt werden (’still activities’) oder jedoch auch über eine bestimmte räumliche Strecke (’move activities’). Ein Beispiel für eine lokale Aktivität wäre z.B. ’Essen in einem Restaurant’. ’Wandern’ hingegen kann nicht auf einen bestimmten Ort reduziert werden. Da die bestehende Simulation ALPSIM nur räumlich punktuelle Informationen entgegennehmen kann, ist eine Aktivität, die sich über einen bestimmten Raum erstreckt, durch Wegpunkte charakterisiert. Solche räumlich ausgedehnten Aktivitäten sind von einer punktuellen Start- und Endaktivität umschlossen. Es wird davon ausgegangen, dass sich die beiden Aktivitätstypen in einem Aktivitätenplan abwechseln. Start- und Endaktivität eines Aktivitätenplans sind per Definition punktuelle Aktivitäten. Ein Beispiel für eine solche Aktivitätenfolge ist:

Hotel - wandern - essen - wandern - trinken - wandern - ausruhen und schlafen

Im unvervollständigten Aktivitätenplan kann neben der Aktivitätendauer auch eine Identität eines Objektes angegeben werden. Dabei handelt es sich um einen sogenannten Via-Punkt. Der Agent versucht dann diesen Punkt zu besuchen und dabei die entsprechende Aktivität auszuführen. Voraussetzung ist, dass der Agent die restlichen Bedingungen ebenfalls erfüllen kann. Die Aktivität wird sonst übersprungen. Die erwartete Bewertung ist die Belohnung des Agenten, falls die Aktivität ausgeführen wird.

```
<plan agentID="44" >
    <act name="hotel" location_type="object_id" id="461" />
    <act name="hike_start" location_type=„object_id" id="461" />
    <act name="hike_waypoint" location_type="node_id" id=„4657" />
    <act name="hike_waypoint" location_type="node_id" id=„4354" />
    <act name="hike_waypoint" location_type="node_id" id=„2333" />
    <act name="hike_waypoint" location_type="node_id" id=„4533" />

    ...

    <act name="hike_waypoint" location_type="node_id" id=„4765" />
    <act name="hike_waypoint" location_type="node_id" id=„4333" />
    <act name="hike_waypoint" location_type="node_id" id=„3234" />
    <act name="hike_waypoint" location_type="node_id" id=„5445" />
    <act name="hike_waypoint" location_type="node_id" id=„3444" />
    <act name="hike_waypoint" location_type="node_id" id=„2123" />
    <act name="hike_waypoint" location_type="node_id" id=„4554" />
    <act name="hike_waypoint" location_type="node_id" id=„3429" />
    <act name="hike_end" location_type=„object_id" id="462" />
    <act name="hotel" location_type="object_id" id="462" />
</plan>
```

Abbildung 3.2: Die Aktivtätenliste entspricht derjenigen aus Abbildung 3.1, jedoch nachdem sie vom Agenten mit räumlichem Wissen aus der mentalen Karte vervollständigt wurde. Der Simulation werden die Wegpunkte zurückgegeben, mit denen der Agent die verschiedenen Aktivitäten räumlich definiert. Aktivitäten mit räumlicher Ausdehnung werden aufgeteilt in 'start', 'waypoint', 'end'

Abbildung 3.2 gibt ein Beispiel eines vom Agenten mit räumlichen Informationen vervollständigten Aktivitätenplans wieder. Dieser wird dann von dem Agenten, d.h. der Simulation ausgeführt. Der Agent vervollständigt folgende Informationen einer Aktivität mit Hilfe seines mentalen Wissens, falls diese nicht schon gegeben sind:

- Aktivitätentyp
- erwartete Dauer
- erwartete Bewertung
- Wegpunkte (>0)
- erwartete Startzeit
- erwartete Endzeit
- Modus ('still' oder 'move')

Falls es dem Agent nicht möglich ist, eine bestimmte Aktivität mit Einhalten der Bedingungen des ursprünglichen Aktivitätenplans mit räumlichem Wissen anzureichern, wird die Aktivität übersprungen. Beim Vervollständigen des Aktivitätenplans hat das Erreichen des Ziels höchste Priorität.
Der Agent kommuniziert mit der Simulation die Wegpunkte der verschiedenen Aktivitäten. Ein Wegpunkt kann dabei auf drei verschiedene Arten definiert werden: Als visuelles Objekt, als Knotenpunkt oder als Koordinate. Mit Angabe des Typs kann zwischen den einzelnen Wegpunktarten unterschieden werden. Die Typen sind 'object_id' für ein Objekt, 'node_id' für einen Knotenpunkt der Route und 'coord' für eine Koordinatenangabe. Die mentale Karte verwendet vorläufig Wegpunkte der ersten zwei Typen.

Anmerkung: Es ist anzumerken, dass eine solche Aktivitätenliste nicht einem Routing entspricht. Durch Miteinbezug von Informationen mit unterschiedlich grosser Unsicherheit ist die Route unvollständig. Dies wird durch die Verwendung von z.B.indirekt wahrgenommenem Wissen deutlich. Der Agent weiss, z.B. dass ein Objekt existiert, jedoch nicht wie er dorthin kommt. Bei der Ausführung wird das fehlende Wissen durch die ausführende

Simulationskomponente ergänzt. Diese hat totales Wissen über die Agentenwelt. Für den Agenten entspricht diese Situation dem Nachschauen auf einer Wanderkarte.

3.3 Planungsalgorithmen

Der Planungsalgorithmus ist neben der visuellen Extraktion von Objekten aus einer Szene der rechenaufwendigste Teil der ganzen Simulation. Ziel ist, einen Aktivitätenplan der bestimmte Tagesaktivitäten beschreibt, anhand des räumlich Wissen des Agenten auszuführen. Das Problem formuliert sich dabei folgendermassen:

> *Gegeben sei eine Datenstruktur aus Objekten, verbunden durch gewichtete Kanten. Sind die Objekte effektiv vom Agenten begangen worden, entspricht die Kante einer Route. Eine Route hat eine bestimmte Richtung. Falls ein Objekt nur visuell wahrgenommen wurde, ist die Kante ungerichtet. Man spricht von einem visuellen Link. Das Ziel ist nun einen Weg vom Start- zum Zielobjekt, der eine möglichst gute Bewertung hält und die zeitlichen Rahmenbedingungen der Aktivitäten bestmöglich erfüllt zu finden. Die Bewertung setzt sich aus der erwarteten Bewertung der verwendeten Kante der mentalen Karte und der im Aktivitätenplan gegebenen Bewertung für die Erfüllung einer Aktivität zusammen.*

Der Aktivitätenplan ist aus einander abwechselnden punktuellen und räumlich ausgedehnten Atkivitäten zusammengesetzt. Die Aufgabe ist, ein Objekt zu finden, welches die nächste punktuelle Aktivität erfüllt und via eine räumliche ausgedehnte Aktivität erreicht wird. Dabei soll die bestmögliche Variante gesucht werden bezüglich Bewertung, Einhalten der Zeit und allenfalls vorgegebenen Orten. Nicht erreichbare punktuelle Aktivitäten werden aus dem Aktivtätenplan gestrichen. Die dann aufeinanderfolgenden räumlich ausgedehnten Aktivitäten zusammengefügt.

3.3.1 Die Routenplanung zwischen zwei lokalen Aktivitäten

Möchte ein Agent von einer punktuellen Aktivität zur nächsten gelangen, muss er einen bestimmten Weg im Raum zurücklegen. Das Zurücklegen des Weges beschreibt eine Aktivität, die eine bestimmte räumliche Ausdehnung hat. Wegpunkte (Via-Objekte) beschreiben diese Aktivität oder eben den Weg.
Für die Wahl des Weges ist die individuelle Bewertung der einzelnen Routenobjekte oder visuellen Links in der mentalen Karte und eine Zeitvorgabe zum Zielobjekt von zentraler Bedeutung. Die Güte der Zeitabschätzung variiert je nach dem Grad der Unsicherheit über einem bestimmten Routenobjekt oder visuellen Link. Der Agent hat von einer Strecke, die er schon mal begangen hat eine recht genaue Abschätzung. Hingegen bei einer Strecke zu einem Objekt, das nur visuell wahrgenommen wurde ist die Zeitangabe eine reine Schätzung. Sie beruht auf einer ungefähren Distanz und einer angenommenen Durchschnittsgeschwindigkeit des wandernden Agenten.
Dem Agenten stehen folgende Verbindungstypen zweier Objekte zur Verfügung:

1. direkt wahrgenommene Routen (kleine Unsicherheit: Route begangen, Masse recht genau)

2. direkt wahrgenommene Routen rückwärts (mittlere Unsicherheit: Route begangen, Masse ungenau)

3. visuelle Objekte (grosse Unsicherheit: Route unbekannt, Masse sehr ungenau)

Je mehr sich ein Agent an direkt wahrgenommene Routen hält, desto kleiner ist das Risiko, dass ein Weg nicht mit der Realität übereinstimmt. Der Nachteil ist, dass so nur wenige

```
findSubroutes( String startID, String endType, long timelimit)
{
        1) init dijkstra network
                for each node in the mental map do:
                        successor       = null
                        time            = 0
                        score           = - infinity

        2) perform Dijkstra to find routes with maximal score and valid time constraint

                Collection visited_nodes;
                Collection expand_candidates;
                Collection neighbours;

                current_node            = start_node
                current_node.score      = 0
                current_node.time       = 0
                current_node.successor  = null
                expand_candidate        = null
                expand_candidates       = {current_node}

                while(expand_candidates not empty)
                {
                        neighbours  = Find all route links connected to the start node.
                                      Find all visual links contained in thel views detected on the previous route link and the start node.
                                      Get a time and score estimation when using a link from the current start node.
                        for all neighbours do
                        {
                                if(current_node.time + neighbour.time < timelimit) AND (neighbour ∈ visited_nodes)
                                {
                                        if(neighbour contained in expand_candidates)
                                                expand_candidate = contained node
                                        else
                                        {
                                                expand_candidate = neighbour node
                                                expand_candidates.add(expand_candidate)
                                        }
                                        if(contained_candidate.score < current_node.score + neighbour.score)
                                        {
                                                expand_candidate.time = current_node.time + neighbour.time
                                                expand_candidate.score = current_node.score + neighbour.score
                                                expand_candidate.successor = current_node
                                        }
                                }
                        add current node to visited_nodes
                        current_node = expand candidate with maximal score and not yet visited
                }

        3) find all objects in the mental map that are a subtype of the end type according to the generalisation knowledge:

                object_candidates = getObjectCandidates(endType)

        4) extract the routes if available:

                Collection results

                for all object_candidates do:
                        if(mentalmap.objectCandidate.successor not null)
                        {
                                time = object_candidate.time
                                score = object_candidate.score weighted with the deviation from the expected time
                                nodes = backtrack to start element via successors and add nodes
                        }
                        results.add( {time, score, nodes} )
        5) return results
}
```

Abbildung 3.3: Der Algorithmus zeigt, wie verschiedene Routen aus der mentalen Karte extrahiert werden. Es handelt sich dabei um eine modifizierte Form von Dijkstras 'shortest path finding'. Die Information in der mentalen Karte wird als ein gerichteter Graph interpretiert. Der Initialknoten ist per Definition gegeben. In einem Expansionsschritt werden folgende Parameter berechnet: 'time' (bisher benötigte Zeit), 'score' (bisher erreichte Bewertung) und 'successor' (Vorgängerknoten). Die Bewertung wird maximiert indem der Knoten der den grössten Bewertungszuwachs verspricht expandiert wird. Ein Knoten wird nur expandiert, falls der Agent im erlaubten Zeitkorridor bleibt und auf dem Knoten eine bessere Bewertung als bisher erreicht wird. Der Zeitkorridor ist definiert als bevorzugte Zeit plus-minus eine erlaubte Abweichung. Nach Beendigung von Dijkstra kann die beste Route gefunden werden indem vom Zielknoten aus den 'successor'-Knoten gefolgt wird. Falls der Zielknoten von Dijkstra nicht erreicht wird, existiert keine Route, welche die Aktivitätsbedingungen erfüllt.

neue Information in die mentale Karte gelangen. Die Chancen auf eine Vergrösserung des mentalen Wissens und damit eine bessere Route in Zukunft zu finden sind klein.
Falls der Agent das Wissen in der mentalen Karte vergrössern möchte, muss er unsichere Informationen verwenden. Beim Ausführen des Planes wird der Agent genaue Weginformationen erlangen, um das entsprechende Objekt zu erreichen. Dieses Wissen kann in der weiteren Routenplanung eingesetzt werden. Der visuelle Link wird deaktiviert für die Routenplanung, falls er einmal benutzt wurde.
In der Simulation soll sich der Agent möglichst am direkt wahrgenommenen Wissen orientieren. Unsichere Informationen werden nur verwendet, wenn keine Alternative besteht oder sie dem Agenten einen entscheidenden Vorteil bringt in der weiteren Routenplanung. Die Selektion der Informationen aus der mentalen Karte bei der Routenplanung, wird über deren Bewertung gesteuert. Ein visueller Link hat eine Bewertung, die negativ proportional zum geschätzten Abstand ist. Die Bewertung einer Route ist positiv. Da der Agent die Bewertung maximieren möchte, wird er versuchen, den kürzesten visuellen Link zu einem Objekt zu nehmen. Die Bewertung eines visuellen Links wird wie die Bewertung einer Route auf eins normiert. So bleiben die verschiedenen Bewertungen konkurrenzfähig. Mit Konkurrenzfähigkeit ist gemeint, dass der Agent die Chance hat, eine negative Bewertung durch eine darauf folgende positive Bewertung wieder wett zu machen. Ein Beispiel ist, dass der Agent nur via einen visuellen Link eine besonders schöne Route findet. Die genaue Berechnung der Bewertung ist in Gleichung 3.1 gegeben.

$$score = -(1 - \frac{1}{distanceEstimation}) \cdot score, \tag{3.1}$$

wobei der $score$ der Bewertung des visuellen Links entspricht und die $distanceEstimation$ der geschätzten Distanz zu dem visuell wahrgenommenen Objekt. Die maximale Bewertung ist eins. Die Normierung ist nicht linear.

Werden direkt wahrgenommene 'Routen rückwärts' in die Routenplanung miteinbezogen, wird deren Bewertung mit einem Gewicht $\in [0, 1]$ multipliziert. Das Gewicht ist konstant und wird bei der Initialisierung der Simulation festgelegt. Der Agent bevorzugt so Vorwärtsrouten mit geringerer Unsicherheit.

Anmerkung Visuelle Links, die den Anfang auf einem Routenobjekt haben, werden auf den Startknoten der Route gesetzt. Nehmen wir an, ein Objekt ist sehr nahe an einer dem Agenten bekannten Strasse. Die beiden Verzweigungen, die das Routenobjekt umschliessen, sind nicht die nächsten zum Objekt. Damit hat der Knoten, mit dem das Routenobjekt beginnt, die beste Bewertung. Der Agent nimmt jedoch nicht zwingend diesen Knoten. Die Wahl hängt im Wesentlichen davon ab, wie hoch die Bewertung zum Startobjekt des visuellen Links ist. Eventuell lohnt es sich einen schlechteren Link zu nehmen. Da jeder visuelle Link nur einmal genommen werden kann, lernt der Agent mit der Zeit, welche Route zum gesehenen Objekt optimal ist.

Der Algorithmus, der zur Planung des Weges verwendet wird ist eine abgewandelte Version von Dijkstras kürzester Pfad Algorithmus. Der Algorithmus hat den Vorteil, dass er direkt auf der Datenstruktur angewandt werden kann. Diese kann als gerichteter Graph betrachtet werden (vgl. Einführung Sektion 3.3)
Dijkstra soll eine Route berechnen, die sich möglichst exakt an dem Zeitvorschlag der Aktivität orientiert und dabei die Summe der internen Bewertungen der einzelnen Elemente der Route maximiert. Je mehr die Zeit abweicht vom initialen Vorschlag, desto negativer soll die Auswirkung auf die Bewertung sein. Die Zeitabweichung bezieht sich auf den gesammten Weg zwischen zwei Objekten. Dijkstra konzentriert sich jedoch nur auf den nächsten Wegpunkt. Es entsteht ein Trade-Off von 'Zeit einhalten' und 'Bewertung maximieren'.

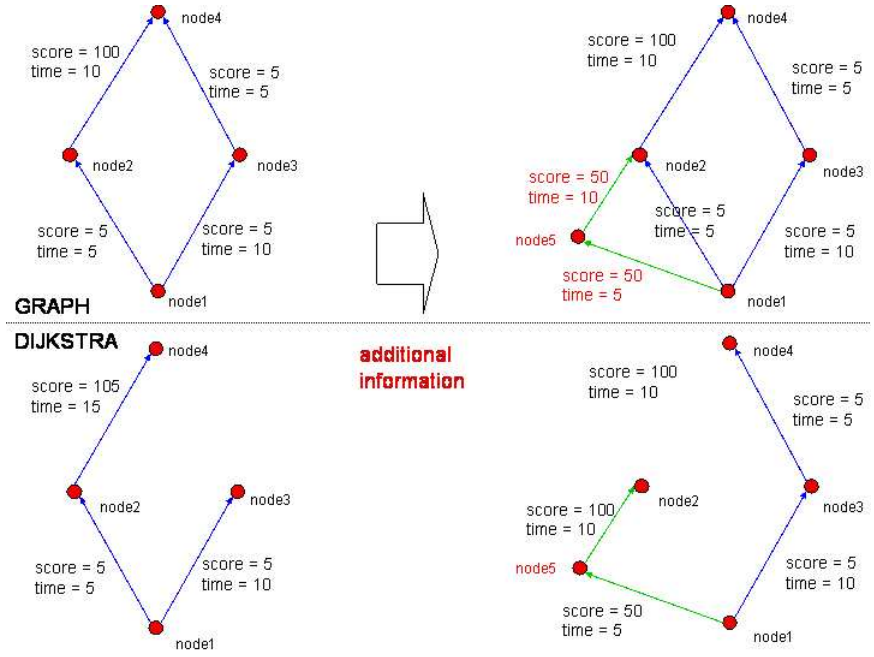


Abbildung 3.4: In der Simulation wird zum Finden einer Route Dijkstra verwendet. Die Route soll sich neben der Maximierung der Bewertung möglichst exakt am Zeitvorschlag der Aktivität orientieren. Die Route darf nur eine bestimmte Abweichung von dieser Zeitlimite haben. In Dijkstra wird deshalb nur zu einem Objekt expandiert, falls die bisherige Route die maximale zeitliche Abweichung nicht überschreitet. Übersteigt die Zeitabweichung jedoch das erlaubte Limit, kann es sein, dass auch eine schlechtere Route, die besser auf das Zeitlimit passt, nicht gefunden wird. Dieser Fall tritt ein, falls die Routen bis zum Überschreiten des Zeitlimits gemeinsame Knoten besitzen. Im Beispiel in der Abbildung soll eine Route von 'node1' nach 'node4' gefunden werden mit einem maximalen Zeitlimit von 20 Minuten. Die gefundene Route hat eine Bewertung von 105 und einen Zeitaufwand von 15 Minuten. Wird nun Information hinzugefügt, wie rechts auf der Abbildung aufgezeigt, fällt die Route dem Zeitlimit zum Opfer. Gäbe es keine alternative Route via 'node3', würde keine Route gefunden. Die alte bessere Route geht verloren.

Konkret wurden zwei ähnliche Algorithmen ausgetestet, die sich dem Problem widmen. Der erste Algorithmus expandiert zu einem Knoten, falls sich dessen Bewertung verbessert und eine maximale Abweichung der bevorzugten Zeit nicht überschritten wird. Die Nachteile der Strategie sind leicht erkennbar. Eine längere Route mit hoher Bewertung wird bevorzugt. Übersteigt die Zeitabweichung jedoch das erlaubte Limit, kann es sein, dass auch eine schlechtere Route, die besser auf das Zeitlimit passt, nicht gefunden wird. Dieser Fall tritt ein, falls die Routen bis zum Überschreiten des Zeitlimits gemeinsame Knoten besitzen. Abbildung 3.4 veranschaulicht das Problem. Die zu lange Route besetzt diese Knoten. Der Algorithmus ist in Abbildung 3.3 gegeben. Die Bewertung der Route wird proportional zur zeitlichen Abweichung gewichtet.
Der A*-Algorithmus scheint auf genau das erwähnte Problem zugeschnitten zu sein. Es handelt sich um eine Erweiterung von Dijkstra. A* expandiert Knoten nicht mehr nur aufgrund einer bisherigen Bewertung, sondern aufgrund der bisherigen Bewertung und einer Abschätzung der Bewertung vom Knoten zu dem expandiert wurde bis zum Zielknoten. Angewandt auf das Zeitproblem ist die geschätzte Zeit, um den Zielknoten zu erreichen, die Zeit, die benötigt wird, um den zu expandierenden Knoten zu erreichen plus eine Abschätzung der Zeit um das Ziel zu erreichen. Es müsste dann die Bewertung expandiert werden, die gewichtet mit der kleinsten geschätzten Abweichung zur bevorzugten Zeit für die Aktivität am grössten ist. Die Idee ist in Gleichung 3.2 illustriert.

$$score_{new} := score_{old} \cdot (1 - \frac{|time_{proposed} - time_{route}|}{time_{proposed} \cdot deviation_{max}}), \tag{3.2}$$

wobei $score_{old}$ die bisherige Bewertung ist, $score_{new}$ die gewichtete Bewertung, $time_{proposed} - time_{route}$ die geschätze Abweichung, falls via einen bestimmten Knoten gegangen wird und $time_{proposed} \cdot deviation_{max}$ der maximalen erlaubten Zeitabweichung entspricht. Ist die geschätzte Abweichung grösser als die maximale erlaubte Abweichung wird nicht expandiert.

Das grosse Problem von A* ist, eine gute Abschätzung für die Zeit zu finden. Ist dies nicht gewährleistet, können die Bewertungen völlig fasch interpretiert werden.

3.3.2 Routenplanung mit Dijkstra - zeitlimitierte Expansion

Abbildung 3.3 gibt den genauen Algorithmus, der verwendet wurde wieder. Es gibt grundsätzlich drei Phasen: Eine Initialisierungsphase, eine Aufbauphase und eine Extraktionsphase. In der Aufbauphase werden die alle Knoten des Graphen initialisiert. Für jeden Knoten wird die bisherige Zeit und Bewertung auf null gesetzt. Der gespeicherte Vorgängerknoten wird gelöscht. In der Aufbauphase wird ausgehend vom Startknoten die Bewertung und die Zeit zum Knoten, zu dem expandiert wurde, propagiert. Expandiert werden nur Knoten, die noch nie besucht wurden, die Zeitbedingung erfüllen und einen Zuwachs an Bewertung garantieren. Jeder Knoten, zu dem expandiert wird, merkt sich den Vorgänger. Das Zeitlimit hat keinen Einfluss auf das Erreichen des Zielknotens. Die Expandierung verläuft vollständig nach einem 'greedy'-Prinzip. In jeder Iteration wird jeweils der Knoten mit der besten Bewertung expandiert. Wurde der Zielknoten erreicht, kann die Route zurückverfolgt werden, indem man den gespeicherten Vorgängern folgt, bis der Startknoten erreicht wird.

Anmerkung: In Untersektion 2.4.1 wird die Bewertung eines Routenobjektes abhängig von der Tageszeit definiert. Die Attraktivitäten die von ALPSIM empfangen werden, sind zu jeder Tageszeit an einem Ort gleich. Wie in Untersektion 2.4.1 beschrieben, kann die Zeitabhängigkeit deshalb genauso als vernachlässigt betrachtet werden.

3.3.3 Routenplanung mit dem A*-Algorithmus - zeitlimitierte Expansion

Der A*-Algorithmus geht hier davon aus, dass die Zeit, um ein Ziel zu erreichen, aus einer exakten Abschätzung bis an eine bestimmten Position ausgeht und einer ungefähren Abschätzung von diesem Punkt zum Ziel. Gleichung 3.3 gibt den Sachverhalt wieder.

$$time_{total} = time_{exact}^{start \to node_i} + time_{estimation}^{node_i \to end}, \tag{3.3}$$

wobei $time_{total}$ die Zeit vom Startpunkt ins Ziel ist, $time_{exact}^{start \to node_i}$ die bekannte Zeit vom Startpunkt bis zum Knoten i und $time_{estimation}^{node_i \to end}$ eine Abschätzung für die restlich benötigte Zeit bis ins Ziel.

Expandiert Dijkstra zu einem Knoten i, zu dem schon einmal expandiert wurde, ist die Zeit den Knoten zu erreichen verschieden von der bisherigen. Die geschätze Zeit, um von Knoten i zum Zielknoten zu gelangen, ist gleich. Durch die unterschiedlichen Abweichungen der gesamten Zeitabschätzungen lassen sich Gewichte für die bisherige Bewertung berechnen. Die entsprechenden Formeln für die Gewichte sind in Gleichung 3.4 und 3.5

gegeben. Je kleiner die Abweichung, desto grösser das Gewicht. Der Knoten wird aktualisiert falls die Gewichtete neue Bewertung grösser ist als die bisherige. Gleichung 3.6 wiedergibt das Selektionskriterium.

$$weight_{new} = \frac{time_{wanted}^{start \rightarrow end}}{|time_{exact}^{start \rightarrow node_i} + time_{exact}^{node_i \rightarrow node_j} + time_{estimation}^{j \rightarrow end} - time_{wanted}^{start \rightarrow end}|}, \tag{3.4}$$

wobei $weight_{new}$ das Bewertungsgewicht der Route via Knoten i ist, $time_{wanted}^{start \rightarrow end}$ die vorgegebene Zeit der Aktivität, $time_{exact}^{node_i}$ die Zeit bis zum Knoten i aufgrund der Angaben in der mentalen Karte, $time_{exact}^{node_i \rightarrow node_j}$ die Zeit für das expandierte Routenstück und $time_{estimation}^{j \rightarrow end}$ die Zeitabschätzung bis ins Ziel.

$$weight_{old} = \frac{time_{wanted}^{start \rightarrow end}}{|time_{exact}^{start \rightarrow node_j} + time_{estimation}^{j \rightarrow end} - time_{wanted}^{start \rightarrow end}|}, \tag{3.5}$$

wobei $weight_{old}$ das Bewertungsgewicht der Route via Knoten j ist, $time_{wanted}^{start \rightarrow end}$ die vorgegebene Zeit der Aktivität, $time_{exact}^{node_j}$ die Zeit bis zum Knoten j aufgrund der Angaben in der mentalen Karte und $time_{estimation}^{j \rightarrow end}$ die Zeitabschätzung bis ins Ziel.

$$(score^{start \rightarrow node_i} + score^{node_i \rightarrow node_j}) \cdot weight_{new} \cdot > (score^{start \rightarrow node_j}) \cdot weight_{old} \tag{3.6}$$

In der Simulation wird die Zeitabschätzung berechnet indem der Dijkstra-Algorithmus rückwärts laufen gelassen wird. Damit erhält man eine Zeitabschätzung für alle Knoten. Der Aufwand, eine Route zu finden mit bekannter Zielidentität, ist doppelt so gross. Ist nur der Typ der Zielaktivität bekannt, muss Dijkstra für jeden Kandidaten einmal zusätzlich ausgeführt werden. Die Laufzeit, um eine Teilroute zu finden, ist demnach, mal die Anzahl Kandidaten für eine Aktivität grösser. Ist die Abschätzung schlecht, versagt der Algorithmus wie schon vorhin. Die Gewichtung ist dann vollkommen falsch. Problematisch ist, dass fast alle Algorithmen, die einen Pfad suchen auf einem Graphen, von einer Minimierung oder Maximierung der Bewertung ausgehen. Ein Pfad zu finden, der genau eine bestimmte Zeit benötigt, ist viel schwieriger.

3.3.4 Die Aktivitätsplanung

Die Aktivitätsplanung beruht auf den Möglichkeiten, eine nächste Aktivität in einem Plan zu erreichen. Der Anfrageplan beinhaltet dabei nur minimale Informationen (Sektion 3.2). Für jede Aktivität soll mindestens der Typ angegeben werden und eine Zeitvorgabe. Für die Startaktivität ist ein Ort definiert. Jeder Aktivität kann anfangs eine Bewertung zugewiesen werden. Diese wird bei einer Integration der Aktivität in den schlussendlichen Aktivitätsplan zur Bewertung der verwendeten Informationen der mentalen Karte dazugezählt.
Die höchste Priorität eines Planungsalgorithmus ist, dass der Agent die Zielaktivität erreicht. Im Aktivitätenplan wechseln sich punktuelle Aktivitäten mit Aktivitäten mit räumlicher Ausdehnung ab. Die erste und letzte Aktivität ist punktuell. Der Agent startet und endet somit an einem eindeutig definierten Ort.
Ein Router, der auf dem Wissen der mentalen Karte und deren Bewertung arbeitet, vervollständigt die räumlich ausgedehnten Aktivitäten mit den entsprechenden Wegpunkten (Sektion 3.3.1). Um eine nächste punktuelle Aktivität zu erreichen, werden zuerst alle möglichen Objekte, auf denen sich die entsprechende Aktivität ausführen lässt, aus der mentalen

```
public void searchRoute(        ...        ) {

        if(plan.activityList.length not reached)  {
                if(move activity) {
                        1) find start and end point or end point type of the move acitvity
                        2) get route collection obeying the time constraint
                        3) if(resulting collection != null) {
                                expand the resulting objects one after the other and update the activity
                                values (expand recursively)
                           } else
                                skip activity
                        4) return if all objects were expanded
                } else {                                // still activity
                         if (last element was a still activity)
                                if(last found activity object contains also this activity type) {
                                        1) update activity values
                                        2) execute next activity in new recursion step
                                        3) return
                                } else {
                                        1) insert a move activity and  update time constraints
                                        2) execute activity in a new recursion step
                                        3) return
                                }
                        } else {
                                1) update activity values and proceed recursively to the next actiivty
                                2) return
                        }
                }
        } else {
                1) send new route
                2) return               // causes search of new routes
        }
}
```

Abbildung 3.5: Der Algorithmus für die Aktivitätsplanung ist rekursiv. Er setzt voraus, dass dem Agenten ein unvollständiger Aktivitätenplan übergeben wird. Dieser wird mit dem räumlichen Wissen in der mentalen Karte komplettiert. Für jede Aktivität muss mindestens der Aktivitätentyp und ein Zeitvorschlag gegeben sein. Falls es sich um die Anfangsaktivität handelt muss zusätzlich der Ort bekannt sein, an dem sie ausgeführt werden soll. Es wird unterschieden zwischen punktuellen und räumlich ausgedehnten Aktivitäten. Der Algorithmus versucht ausgehend von der punktuellen Startaktivität via die Aktivität mit räumlicher Ausdehnung die nächste punktuelle Aktivität zu erreichen. Dazu werden die Routen zu sämtlichen Objekten untersucht, auf denen sich die punktuelle Aktivität ausführen lässt. Existiert eine Route, die die Zeitbedingung erfüllt, wird zur nächsten punktuellen Aktivität expandiert. Der Algorithmus versucht rekursiv möglichst schnell eine Lösung zu finden. Diese muss nicht optimal sein. Rekursiv werden alle möglichen Lösungen durchsucht.

Karte des Agenten extrahiert. Dazu wird das Generalisationswissen (Abbildung 2.13 in Sektion 2.3) verwendet. Gesucht sind alle Objekte, deren Typ den aktuellen Aktivitätentyp als Supertyp haben. Falls eine Route zu einem der extrahierten Kandidaten innerhalb des vorgegebenen Zeitlimits existiert, wird diese rekursiv expandiert. Der Agent versucht nun, ein geeignetes Objekt für die nächste punktuelle Aktivität zu finden. Andernfalls wird das Objekt ignoriert. Der Algorithmus traversiert so die im Aktivitätenplan gegebene Liste von Aktivitäten. Die gefundenen Gesamtrouten werden nach ihrer Bewertung geordnet zwischengespeichert.

Es bestehen verschiedene Strategien, welche der gefundenen Gesamtrouten an die Simulation weitergegeben werden soll. Idealerweise ist es diejenige, die nach dem Algorithmus die höchste Bewertung erhalten hat. Aufgrund der in Kapitel 3.3.1 beschriebenen Problemen handelt es sich nicht um die optimale Route, die auf Grund des mentalen Wissens möglich wäre. Es handelt sich um die beste Route, die der Algorithmus gefunden hat. Dies lässt sich mit dem Verhalten eines Individuums vereinbaren. Oft orientiert sich ein solches nicht nach der optimalen und effizientesten Route. Eine weitere Variante wäre, einfach die erst beste Route verwenden. Die Route wäre dann höchstwahrscheinlich schlechter in der Bewertung, der Algorithmus aber im Durchschnitt wesentlich schneller. Der verwendete Algorithmus ist in Abbildung 3.5 in der Form von Pseudocode wiedergegeben.

Die Struktur der mentalen Karte sorgt dafür, dass schnell auf die benötigten Daten zugegriffen werden kann. Von einem bestimmten Objekt hat man Zugriff auf sämtliche Routenobjekte, die dort starten. Durch das Routenobjekt ist via die enthaltenen 'Action Links' auch ein schneller Zugriff auf sämtliche visuell wahrgenommenen Objekte gewährleistet.

Trotz dem geordneten Zugriff auf die Daten hat das Problem der Aktivitätsplanung einen sehr hohen Grad an Komplexität. Dieser wird damit reduziert, dass einerseits der Agent nur ein sehr begrenztes Wissen hat. Der Suchraum wird so im Vergleich zu einem totalen Wissen enorm verkleinert. Durch das Vorgeben eines Aktivitätentypes und einer Zeitdauer für jede Aktivität, findet eine weitere Einschränkung statt.
Es ist sehr schwer, all diese immer noch sehr beschränkten Methoden in einen effizienten Algorithmus zu packen. Der vorgestellte Algorithmus hat den Nachteil, dass er im schlechtesten Fall exponentiell ist. Der Exponent ist dabei charakterisiert durch die jeweiligen Objektkandidaten, die bei einer Attraktivität expandiert werden. Dies sind die Kandidaten die innerhalb eines bestimmten Zeitlimits erreichbar sind. Es wird angenommen, dass die Anzahl Objekte, die für eine Aktivität geeignet sind und zu denen eine gültige Route existiert, beschränkt ist. Dasselbe gilt für die Anzahl Aktivitäten, die ein Agent jeden Tag bewältigt. Zusätzlich ist der rekursive Algorithmus so ausgelegt, dass er möglichst schnell eine Route findet. Es wird versucht, mit Hilfe eines bestimmten gültigen Objektkandidaten für eine Aktivität eine Lösung zu finden, bevor der nächste Kandidat überprüft wird.

3.4 Generieren von Tagesaktivitäten

Um den Agenten zu automatisieren, ist es nötig, Tagesaktivitäten zu generieren. Diese werden dann vom Agenten mit dem mentalen Wissen komplettiert. Wir nehmen an, dass das bekannte mentale Wissen keinen Einfluss bei der Wahl eines bestimmten Aktivitätentyps in die Tagesplanung hat.
Der Agent bevorzugt, bestimmte Aktivitäten an bestimmten Tageszeiten auszuführen. Dieses Wissen und die dazugehörigen möglichen Aktivitäten sollen im Agent fest verankert sein. Der Agent plant den Tagesablauf, indem er zuerst die Zeit bestimmt, an dem der Tag für ihn beginnt, und eine Zeit, an dem er sein Ziel erreicht haben will. Dazu wird eine zufällige Abweichung von einer vorbestimmten durchschnittlichen Startzeit und einer durchschnittlichen Endzeit gewählt. Es ist Voraussetzung, dass der Agent im Ort, an dem er sich gegenwärtig befindet, beginnt. Ein Tag wird jeweils mit einer Aktivität 'ausruhen und schlafen' beendet. Diese ist im Generalisationswissen definiert und hat bestimmte Typen von Objekten wie z.B. 'Hotel' als untergeordneten Typ.
Zufällig wird entschieden, wie viele Aktivitäten der Agent während des Tages ausüben soll. Es wird ein vernünftiger Maximalwert gesetzt. Die einzelnen Aktivitäten werden bestimmt, indem zufällig eine Tageszeit gewählt wird. Für jede Aktivität wird die kleinste zeitliche Abweichung berechnet. Den einzelnen Abweichungen werden umgekehrt proportional zu ihrer Grösse Wahrscheinlichkeiten zugeteilt. Die Aktivität, die am nächsten an der zufällig gewählten Tageszeit liegt, hat die grösste Wahrscheinlichkeit, für den Aktivitätenplan gewählt zu werden. Die Summe der Wahrscheinlichkeiten soll Eins sein. Den Aktivitäten wird ein entsprechendes Intervall zwischen Null und Eins zugeteilt. Durch eine weitere Zufallszahl $\in [0, 1]$ wird die definitive Aktivität gewählt. Der Vorgang wird wiederholt, bis alle Tagesaktivitäten bestimmt sind.
Für eine gewählte Tagesaktivität wird die Wahrscheinlichkeit nochmals am selben Tag wiedergewählt zu werden, sinken. So wird verhindert, dass der Agent um zwölf essen geht und um zwei gerade noch einmal. Die Wahrscheinlichkeit, dass eine Aktivität gewählt, wird zusätzlich vom zeitlichen Abstand von der letzen Wahl abhängig gemacht, falls eine solche stattgefunden hat. Der Einfluss nimmt mit grösser werdendem Abstand ab. Die Abnahme sei exponentiell, wobei deren Geschwindikeit für jede Aktivität durch eine gegebene Halbwertszeit definiert ist (Gleichung 3.7, 3.8 und 3.9). Diese ist fest vorgegeben für jeden Aktivitätstyp.

$$probability_i = \frac{deviation_i \cdot w_i}{\sum_{j=1}^{n} deviation_j \cdot w_j}, \tag{3.7}$$

wobei $probability_i$ die Wahrscheinlichkeit ist, dass die Aktivität i gewählt wird, $deviation_i$ die zeitliche Abweichung vom Optimum der i-ten Aktivität und w_i die Gewichtung dieser Abweichung. Die Gewichtung ($\in [0, 1]$) ist Abhängig vom momentanen zeitlichen Abstand an dem die Aktivität das letzte mal an gleichen Tag verwendet wurde. Dabei konvergiert das Gewicht gegen eins exponentiell mit der Zunahme der der Zeit, wie in Gleichung 3.8 dargestellt.

$$w_i = -\frac{1}{\Delta t_i \cdot v_i + 1} + 1, \tag{3.8}$$

wobei Δt_i den zeitlichen Abstand zu dem Zeitpunkt an dem die Aktivität i das letzte mal am gleichen Tag verwendet wurde angibt (0 falls sie noch nie verwendet wurde) und v_i die Geschwindigkeit, mit der das Gewicht gegen den Wert eins konvergiert, falls die Attraktivität i schon einmal am selben Tag verwendet wurde. Diese Geschwindigkeit ist durch die Halbwärtszeit (Gleichung 3.9) der verwendeten Exponentialfunktion bestimmt. w_i wird wieder auf Null gesetzt, falls die Aktivität in den Plan aufgenommen wird.

$$0.5 = (-\frac{1}{\Delta t_i \cdot v_i + 1} + 1) \tag{3.9}$$

wobei Δt_i die gegebene Halbwärtszeit der i-ten Aktivität ist und v_i die zu bestimmende zeitliche Zunahmegeschwindigkeit des Gewichtes w_i.

3.5 Wissensaustausch zwischen den Agenten

Sobald in der Simulation mehrere Agenten mit unterschiedlichem mentalen Wissen agieren, ist es möglich, eine sinnvolle Kommunikation zwischen den Agenten zu etablieren. Das Ziel einer solchen Kommunikation ist ein Wissensaustausch zwischen den Agenten. Der Wissensaustausch soll die mentalen Karte eines Agenten erweitern, ohne selber visuelle Eindrücke sammeln zu müssen. Solche kooperierende Agenten setzen eine gemeinsame Semantik des Wissen voraus. Dies ist hier Teilweise gegeben. Die Definitionen von Objekten, sowie die Struktur, wie das Wissen aufgebaut ist, ist bei allen Agenten gleich. Die Informationen der Agenten unterscheiden sich jedoch in deren Bewertung. Deshalb ist es nicht sinnvoll, fremde Informationen mit den zugehörigen individuellen Bewertungen ins eigene Wissen einzubauen, ohne sie auf die eigene Nützlichkeit zu prüfen.
Die Agenten in der Simulation können nur miteinander kommuniziern, falls sich diese am selben Ort befinden. Dabei gehen sie nach folgendem Schema vor:

1. Der Agent erkennt die möglichen Kommunikationspartner
2. Der Agent übermittelt die Anfage an die Kommunikationspartner
3. Die Kommunikationspartner übermitteln die persönlichen Antworten mit der persönlichen Bewertung
4. Der Agent beurteilt die Antwort mit der höchsten Bewertung.

Die Anfrage beschränkt sich diesem Fall auf das Senden von einem Aktivitätsplän, der von den fremden Agenten mit räumlicher und zeitlicher Information vervollständigt wird. Die Antwort des angefragten Agenten kann als eine Wegbeschreibung angesehen werden. Sobald diese in der Simulation ausgeführt wird, wird die mentale Karte des Agenten mit den gewonnenen visuellen Eindrücken erweitert. Da der andere Agent unterschiedliche Informationspräferenzen haben kann, ist eine Route mit hoher Bewertung nicht zwingend gut.

Es bestünde die Möglichkeit, die einzelnen Agenten zu bewerten, sobald die Informationen persönlich beurteilt wurde. Eine Bewertung des antworteten Agenten wäre dann hoch, falls der Agent den fremden Aktivitätsplan ausprobiert und eine ähnliche Bewertung dafür sammeln kann. Ist die Bewertung verschieden, kann der fragende Agent davon ausgehen, dass der angefragte Agent entweder unterschiedliche Informationspräferenzen hat, oder der Plan schlecht war. Letzteres ist der Fall, wenn die Bewertungen der mentalen Karte zu ungenau sind. Durch senken der Bewertung des Agenten wird dieser als ungeeigneter Kommunikationspartner deklariert. Diese Bewertung der antwortenden Agenten ist in der Simulation jedoch nicht implementiert.

3.6 Verwenden von sich gegenseitig beeinflussenden Attraktivitäten

In Sektion 2.4.2 wird beschrieben, wie sich die Attraktivitätstypen gegenseitig beeinflussen. Ein Attraktivitätstyp ist immer abhängig von den anderen Attraktivitätstypen, die der Agent wahrnehmen kann, und zusätzlich von einer unbekannten Attraktivität. Zwei Attraktivitätstypen werden als gegenseitig abhängig betrachtet, wenn der Agent sie innerhalb eines bestimmten zeitlichen Intervalls wahrnehmen kann. Die Grösse des zeitlichen Intervalls ist abhängig von der Häufigkeit, mit der die Attraktivität zuvor wahrgenommen wurde. Je häufiger das Vorkommen, desto grösser das Intervall.
Bei der Routenplanung wählt der Agent die Infomationen anhand von Erfahrungswerten in der mentalen Karte aus, die in ihrer Bewertung wiederspiegelt wird. Nehmen wir an, dass sich bestimmte Informationen in der Agentenwelt spontan ändern, ohne einem bestimmten Muster zu folgen. Bei der Ausführung des Aktivitätenplan aus kann es dann vorkommen, dass die erwartete Bewertung nicht mit der tatsächlich wahrgenommenen übereinstimmt. Hält sich diese Abweichung über einen bestimmten Zeitraum in einem gewissen Ausmass, kann mit einer grossen Wahrscheinlichkeit davon ausgegangen werden, dass diese Attraktivität auch im weiteren Routenverlauf verändert bleibt.
Ausgehend von dieser Tatsache kann ein neuer Aktivitätenplan generiert werden. Dieser versucht der Veränderung Rechnung zu tragen. Dadurch, dass die Attraktivitäten sich gegenseitig beeinflussen, wird eine wahrgenommene Änderung der Attraktivität einer Information nicht nur diese selbst beeinflussen, sondern alle Attraktivitäten von Informationen, die von ihr abhängig sind.
Die veränderte Attraktivität einer Information, kann entweder eine andere Attraktivität verstärken oder abschwächen. Dies ist abhängig davon, ob die Abweichung von der vermuteten Attraktivität aufgrund der mentalen Karte positiv oder negativ ist. Der Sachverhalt wird mit folgendem Beispiel deutlich: Da der Agent die schöne Aussicht bisher nur dann wahrgenommen hat, als die Sonne schien, vermutet er, dass die Attraktivitäten der beiden Informationen voneinander abhängig sind. Jetzt nimmt der Agent beim Ausführen eines Planes entgegen seiner bisherigen Erfahrungen weniger Sonnenschein auf einer Teilroute wahr. Der Agent glaubt, dass die Wahrscheinlickeit für die schöne Aussicht ebenso auf den übrigen Teilrouten zurückgeht. Nimmt er jedoch mehr Sonneschein als erwartet auf einer bestimmten Teilroute wahr, so geht er davon aus, dass er auf den übrigen Teilrouten vermehrt eine Aussicht als schön empfindet.
Durch die damit verbundene Änderung in der Bewertung der verschiedenen Routenobjekte entspricht nun eine andere Route dem Agenten besser. Der Aktivitätenplan bleibt derselbe. Die Änderungen der Informationsbewertungen können ermittelt werden, indem der Agent die Attraktivitätsabweichungen in die Bewertung der Routenobjekte einfliessen lässt. In der Simulation wird in einer solchen Situation eine Gewichtung für die einzelnen Attraktivitätstypen berechnet. Das Gewicht wird mit der bisherigen Attraktivität multipliziert. Dadurch ändert sich die Bewertung der einzelnen Routenobjekte und somit der Gesamtroute. Der neue Plan wird dann generiert und ausgeführt, falls die Abweichung der neuen Bewertung

der restlichen Route von der bisherigen einen gewissen konstanten Schwellenwert übersteigt.
Die Abweichung wird durch eine nichtlineare Funktion bestimmt. Kleine Abweichungen haben ein grösseren Einfluss als sehr grosse Abweichungen. Das Resultat ist eine Art Sättigungseffekt. Lineare Abweichungen könnte die Attraktivitäten zu einseitig gewichten. Eine Veränderung von Null zu irgendeinem auch noch so kleinen Wert, wäre dann eine Abweichung von Unendlich. Die Abweichungsfunktion ist gegeben durch:

$$deviation_{attractivity_i} = \begin{cases} \frac{attractivity_i^{new}}{attractivity_i^{old}} & \text{falls } attractivity_i^{new} < attractivity_i^{old} \\ 2 - \frac{attractivity_i^{old}}{attractivity_i^{new}} & \text{falls } attractivity_i^{new} > attractivity_i^{old} \\ 1 & \text{sonst,} \end{cases} \tag{3.10}$$

wobei $deviation_{attractivity_i}$ die Bewertungsabweichung des aktuellen Routenobjekts für Attraktivität i ist. $Attractivity_i^{new}$ ist die gerade wahrgenommene Bewertung und $attractivity_i^{old}$ die alte Bewertung zum Zeitpunkt der Routengenerierung desselben Attraktivitätstyps auf der gleichen Teilroute. Ist $deviation_{attractivity_i} = 1$ sind die Bewertungen gleich. Bei $deviation_{attractivity_i} = 0$ hat die Bewertung unendlich viel abgenommen. Fass $deviation_{attractivity_i} = 2$ hat sie unendlich viel zugenommen.

Der Gewicht berechnet sich für die Attraktivität folgendermassen:

$$weight_{attractivity_i} = deviation_{attractivity_i} \cdot (\sum_{attractivity_j} P(attractivity_i|attractivity_j) \cdot deviaton_{attractivity_j}), \tag{3.11}$$

wobei $weight_{attractivity_i}$ die Gewichtung der Attraktivität i ist, $P(attractivity_i|attractivity_j)$ die Wahrscheinlichkeit ist, dass die Attraktivität i aktiv ist, fass Attraktivität j wahrgenommen wird (vgl. Gleichung 2.11), falls j wahrgenommen wurde, $deviaton_{attractivity_j}$ die festgestellte Abweichung der Attraktivität j. Es gilt zusätzlich, dass $i \neq j$ und die Abweichung von der unbekannten Attraktivitätskomponente immer gleich eins ist (d.h. $deviation_{unknownattractivity} = 1.0$). Findet keine Veränderung statt ist die Abweichung $deviation_{attractivity_i}$ jeweils eins und somit auch die Gewichtung, da die Summe der Wahrscheinlichkeiten eins ist (vgl. Gleichung 2.9).

Kapitel 4

Testen der Simulation

Im Folgenden werden die in Kapitel 2 und 3 besprochenen Eigenschaften der Simulation anhand von Daten ausgetestet. Darin enthalten sind der Aufbau der Datenstruktur, die Wiederverwendung der gespeicherten Daten für die Aktivitätsplanung und das selbständige Erweitern des räumlichen Wissens. Anfangs wird von einem einzelnen Agenten ausgegangen. Später wird die Simulation auf mehrere Agenten erweitert. Abbildung 4.1 definiert

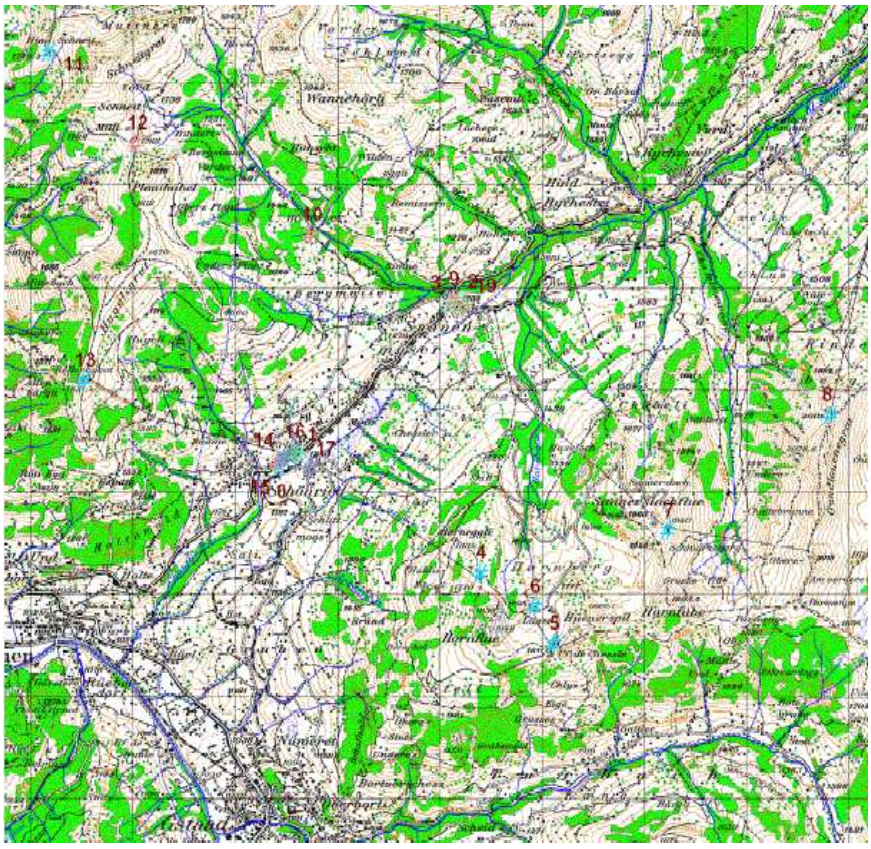


Abbildung 4.1: Die Karte zeigt die geographische Umgebung in der sich der Agent bewegt. Die Objekte, die der Agent besuchen kann, sind mit Nummern beschriftet. Tabelle 4.1 gibt Typinformationen und Identität der referenzierten Objekte wieder. Die Objekte des Strassennetzwerks, die Routenverzweigungen, sind der Übersicht halber nicht eingezeichnet. Dasselbe gilt für die nicht begehbaren Objekte wie Wald.

object id	object type	location x	location y
0	*hotel*	588444.997878	150258.012133
1	*hotel*	588615.761489	150370.22822
2	*hotel*	590186.786712	151863.190078
3	*hotel*	590094.086466	151843.674237
4	*restaurant*	590396.582006	149213.914625
5	*restaurant*	591116.228653	148523.541739
6	*restaurant*	590925.949201	148889.463763
7	*restaurant*	592228.631606	149626.186772
8	*restaurant*	593819.17267	150758.105566
9	*fountain*	590130.678669	151885.1454
10	*fountain*	588735.296017	152519.410241
11	*restaurant*	586179.94055	154295.351798
12	*fountain*	587027.659905	153411.04024
13	*restaurant*	586515.369072	151093.534088
14	*restaurant*	588449.82006	150314.045951
15	*fountain*	588416.371004	150298.993876
16	*information*	588562.710622	150390.142552
17	*store*	588744.06722	150231.364067
18	*store*	588480.655908	150337.982931
19	*information*	590228.107734	151812.615899

Tabelle 4.1: Die Tabelle beinhaltet Objekteidentität, Objekttyp, sowie Koordinaten eines Objektes. Die Identität entspricht auch der Referenznummer in Abbildung 4.1.

die Agentenwelt. Es werden authentische GIS Daten der Region Gestaad verwendet. Alle begehbaren Objekte, die in die Agentenwelt übertragen wurden, sind mit Nummern beschriftet. Die zugehörige Referenz ist in Tabelle 4.1 gegeben. Die Knotenpunkte des Strassennetzwerkes sowie alle nicht begehbaren Objekte wie z.B. 'Wald' sind in der Karte der Übersicht halber nicht eingezeichnet. Jedem Agenten wird ein bestimmtes Initialwissen in der Form von ausgeführten Tagesplänen zur Verfügung gestellt. Die entsprechenden Initialrouten sind in Appendix A.1 abgebildet.
Die Grundeinstellungen der mentalen Karte sowie eine kurze Erklärung der einzelnen Parameter ist in Appendix A.2 gegeben. Diese haben sich beim Herumexperimentieren als sinnvoll herausgestellt. Werden in einem Experiment andere Parameter benutzt, wird dies jeweils explizit angegeben.

Anmerkung: In der mentalen Karte werden keine Koordinaten verwendet. Zur Visualisierung der mentalen Karte werden die Daten der Einfachheit halber in einem Koordinatensystem angezeigt. Die Objektkoordinaten können in einer 'Lookup-table' nachgeschlagen werden (vgl. Tabelle 4.1).

4.1 Aufbau der Datenstruktur

Der Agent legt das wahrgenommene räumliche Wissen in einer speziellen Datenstruktur ab. Eine genaue Beschreibung zur Vorgehensweise ist in Sektion 2.2 gegeben. Die folgenden Experimente testen die verschiedenen Merkmale anhand von realen Daten. Zuerst wird die Informationsselektion getestet, dann wird die Speicherstruktur untersucht.

4.1.1 Erkennen einer Ansicht

Eine Ansicht ist zusammengesetzt aus verschiedenen gleichzeitig wahrgenommenen Objekten. Der Agent beurteilt ein Objekt aufgrund von Merkmalen, wie deren Häufigkeit, Präferenz, Winkel zur Blickrichtung etc.. Diese werden in einer Bewertung umgesetzt, welche die momentane Wichtigkeit des Objektes für den Agenten wiederspiegeln soll. Eine zeitliche Zerfallsfunktion und ein Bewertungszugewinn bei jeder Wahrnehmung des Objektes geben einen Signalcharakter, falls man die Bewertung über ein bestimmtes Zeitintervall betrachtet. An Orten, wo ein Objekt vermehrt wahrgenommen wird, werden Bewertungsaus-

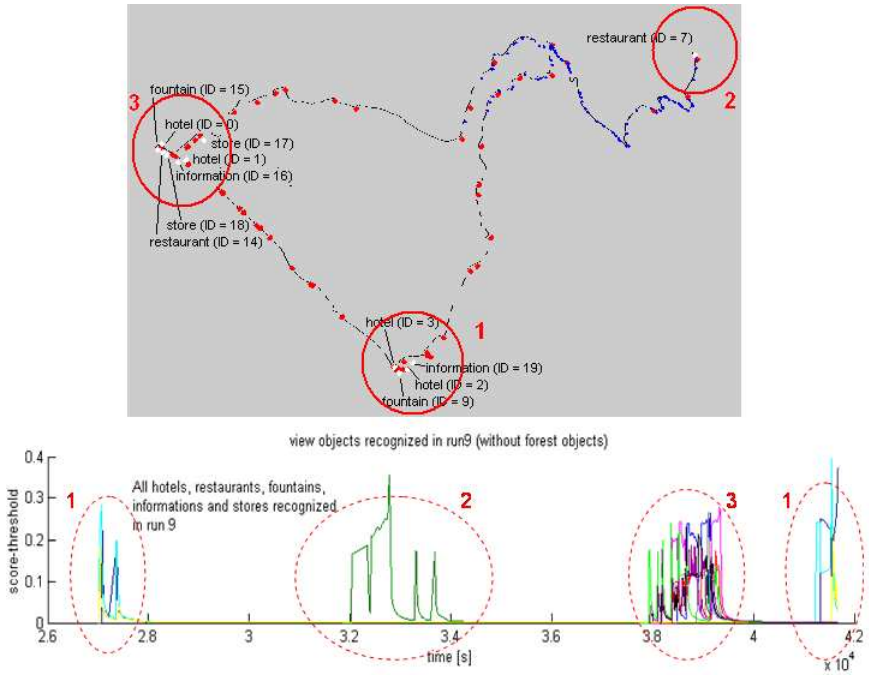


Abbildung 4.2: Oben in der Grafik ist die Route abgebildet, die der Agent abläuft. Die roten Punkte entsprechen dabei Verzweigungen. Die weissen Punkte sind die begehbaren Objekte über welche der Agent visuelle Informationen erhält. Die Objekte konzentrieren sich auf drei verschiedene Orte, die in der Grafik mit einem roten Kreis markiert sind. Blaue Punkte entsprechen Orten, wo eine schöne Aussicht wahrgenommen wird. Im unteren Teil der Abbildung sind die Signale der Objekte gegeben. Man stellt fest, dass sich die Signale auf drei Orte konzentrieren. Da der Agent konstante Geschwindigkeit hat, sind die zeitlichen Abstände proportional zu den Längen der Wegstücke. Die Signalausschläge lassen sich den drei Objektanhäufungen zuordnen. Daraus lässt sich schliessen, dass der Agent die Objekte wahrnimmt, wenn sie genug nah in seinem Gesichtsfeld sind. Sobald sie aus dem Gesichtsfeld geraten, nimmt die Bewertung wieder ab.

schläge erwartet. Die Bewertung eines Objektes wird in einem Kurzzeitgedächtnis gespeichert. Nimmt der Agent das Objekt wahr und ist die gerade aktualisierte Bewertung über einem bestimmten Schwellenwert, so wird das Objekt in die gegenwärtige Ansicht aufgenommen. Dieser Schwellenwert ist adaptiv und berechnet sich aus den Bewertungen aller Objekte im Kurzzeitgedächtnis. Da dieser Speicher nur begrenzte Kapazität besitzt, muss er von Zeit zu Zeit aufgeräumt werden. Alle Objekte mit einer Bewertung unter dem aktuellen Schwellenwert werden gelöscht. Durch die Zerfallsfunktion der Bewertung sind nur die Objekte im Kurzzeitgedächtnis, die die nahe Vergangenheit der Agentenwahrnehmung beschreiben. Der Schwellenwert passt sich diesen Objekten an. Es entsteht eine Kontextabhängigkeit durch die zuletzt wahrgenommen Objekte. Es wird erwartet, dass der Schwellenwert nach einem Bewertungszuwachs durch eine wahrgenommene Ansicht schneller reagiert als der Schwellenwert, da diese sämtliche Objekte des Kurzzeitgedächtnis beinhaltet. Eine komplette Ansicht wird im Langzeitgedächtnis des Agenten abgelegt. Ist die Ansicht bereits vorhanden, wird sie wiedererkannt werden. Eine genaue Beschreibung der Informationsselektion ist in Sektion 2.2 gegeben.

In einem ersten Experiment wird der Signalcharakter der Objekte untersucht. Grundlage dafür bildet Gleichung 2.3 und 2.5. In Abbildung 4.2 unten, wurde die Signalentwicklung visualisiert, während der Agent die Route oben in der Abbildung ausführt. Die einzelnen Signalausschläge sind deutlich zu erkennen. Es wurden nur die Signale von Objekten be-

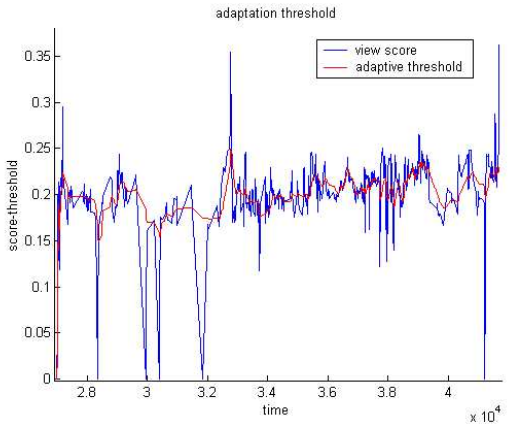


Abbildung 4.3: Die blaue Kurve zeigt die durchschnittliche Bewertung einer Ansicht über die Zeit. Die rote Kurve gibt jeweils den zugehörigen Schwellenwert an. Der Schwellenwert probiert jeweils an die durchschnittliche Bewertung aller Objekte im Kurzzeitgedächtnis zu adaptieren. Die Adaption ist zeitlich verzögert. Die Verzögerung ist in der Abbildung erkennbar. Diese Trägheit des Schwellenwertes kann durch die Grösse des Kurzzeitgedächtnisses gesteuert werden.

trachtet, die in Abbildung 4.1 referenziert sind. Es handelt sich dabei um alle begehbare Objekte, die nicht eine Strasse oder ein Verzweigungsknoten sind. Es gibt drei Signalanhäufungen, die eindeutig den Objektansammlungen in der obigen Grafik entsprechen. Die Signale fallen ab, sobald die Objekte das Sichtfeld des Agenten verlassen. Die Bewertungsentwicklungen der Objekte scheinen korrekt.
In einem weiteren Experiment wird der Schwellenwert untersucht. In Abbildung 4.3 ist der Schwellenwert über einen bestimmten Abschnitt einer Route visualisiert. Es handelt sich dabei um einen Ausschnitt aus derselben Route, die im vorherigen Experiment verwendet wurde. Es ist deutlich erkennbar, wie der Schwellenwert, in der Abbildung rot, dem Signaldurchschnitt der Objekte in der momentanen Ansicht mit leichter Verzögerung folgt. Das Resultat ist eine Glättung der blauen Kurve. Die Verzögerung ist von der vorgegebenen Grösse des Kurzzeitgedächtnis abhängig. Im Experiment wurde die Grösse auf hundert Einträge limitiert. In spätern Experimenten wird ein Schwellenwert von zehn verwendet. Dies hat eine raschere Adaption zur Folge.
Abbildung 4.4 zeigt die Signale des Objekts vom Typ 'Hotel' und der Identität 0 und einem Objekt vom Typ 'Wald' mit der Identität 213. Zusätzlich wurde der Schwellenwert eingezeichnet. Die Daten stammen wieder von der schon vorher verwendeten Route. Die Signale überschreiten den Schwellenwert an einigen Orten. Ist das Signal grösser als der Schwellenwert und wird das Objekt gerade wahrgenommen, so wird es in die aktuelle Ansicht aufgenommen. Sobald die Ansicht komplett ist, gelangt diese ins Langzeitgedächtnis.

4.1.2 Datenreduktion

Das Ziel der Datenreduktion ist, die Menge der Informationen, die der Agent wahrnimmt, zu reduzieren. Nur die signifikanten Informationen sollen weiter verarbeitet werden. Welche Infomationen signifikant sind, hängt vom Verwendungszweck der Daten ab. Für den Agenten sind dies Informationen, die die räumliche Umgebung beschreiben. Später sollen sie dafür verwendet werden, eine Route zu finden, die einen bestimmten Aktivitätenplan erfüllt.
Eine erste Phase der Informationsreduktion findet beim Erkennen von Ansichten statt. Dabei werden nur die Informationen in eine Ansicht aufgenommen, die zur Zeit der Wahrnehmung eine Bewertung haben, die über der momentanen Durchschnittsbewertung der

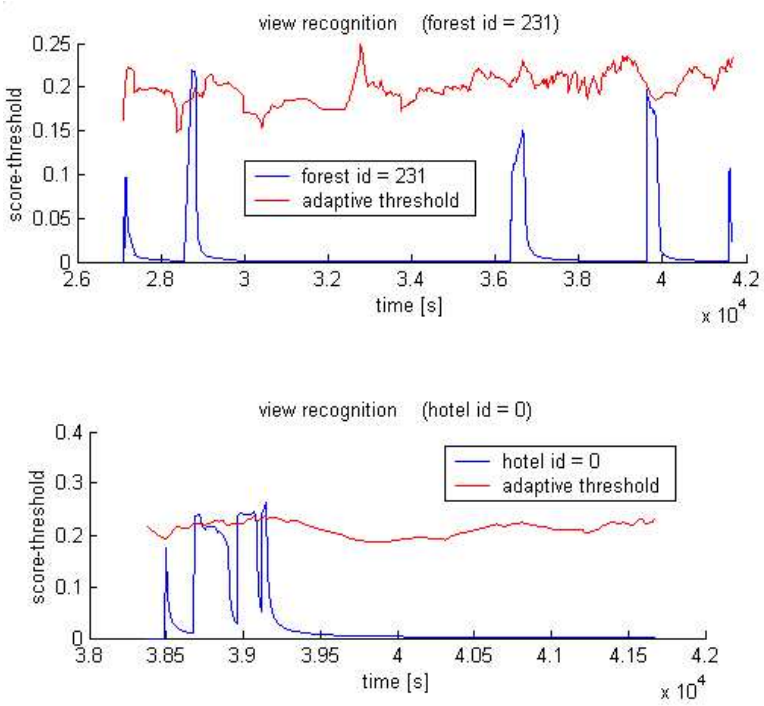


Abbildung 4.4: Die Grafik zeigt die Bewertungskurven von zwei Objekten. Die obere Abbildung gibt das Signal des Objekts vom Typ 'Hotel' und der Identität 0, die untere Abbildung diejenige des Objekts vom Typ 'Wald' mit der Identität 231 wieder. Das Waldobjekt ist rundum sichtbar. Die Ausschläge des Signals verteilen sich über die gesamte Route. Das Hotelobjekt ist nur kurz sichtbar. Das Signal schlägt nur während kurzer Zeit einmal aus. Anscheinend ist das Objekt von anderen Positionen nicht oder zuwenig gut sichtbar. Objekte können sich auch gegenseitig verdecken. An bestimmten Orten übersteigt, die Signalkurve den Schwellenwert. Nimmt der Agent das Objekt dort wahr, so wird es in die aktuelle Ansicht aufgenommen. Die komplette Ansicht wird im Langzeitgedächtnis abgelegt.

Objekte im Kurzzeitgedächtnis ist. Die Simulation sendet in konstantem zeitlichen Abstand die visuellen Daten einer Ansicht. Es kann vorkommen, dass nacheinander eine fast identische Ansicht wahrgenommen wird. Die Information, die sich ein Agent merkt, kann nochmals reduziert werden, indem solche ähnlichen Ansichten zusammengefasst werden. Dabei sind das 'source'-Objekt der Ansicht, die Perspektive und die Ähnlichkeit der Anordnung der gesehenen Objekte ausschlaggebend. Der Agent merkt sich dort, wo ein grosser optischer Fluss entsteht, mehr Objekte. Die Anzahl Ansichten, die sich ein Agent merkt, kann mit einem Schellenwert reguliert werden ('similarity'-Schwellenwert oder Ähnlichkeitsschwellenwert). Hat eine Ansicht ein gleiches 'source'-Objekt, eine gleiche Perspektive und ist die Ähnlichkeit mit der vorherigen Ansicht über dem Schwellenwert, so werden diese verschmolzen.

In einem Experiment soll die Fähigkeit, die Anzahl an Ansichten zu reduzieren, deutlich gemacht werden. Der genaue Algorithmus ist in Sektion 2.2.2 beschrieben. Im Experiment begeht der Agent dieselbe Route mit unterschiedlichem Ähnlichkeitsschwellenwert. Der Agent in Abbildung 4.5 benutzte für die Route im ersten Durchgang einen Schwellenwert der Grösse 0.0 und im zweiten Durchgang einen Wert der Grösse 1.0. Null bedeuted, dass nur aufgrund von 'source'-Objekt und Perspektive entschieden wird, ob zwei Ansichten gleich sind. Der Schwellenwert wird immer überschritten. Es werden nur wenige Ansichten gespeichert. Bei einem Schwellenwert von eins gilt die Ansicht nur dann als gleich, wenn auch die topologische Anordnung der gesehenen Objekte vollständig übereinstimmt.

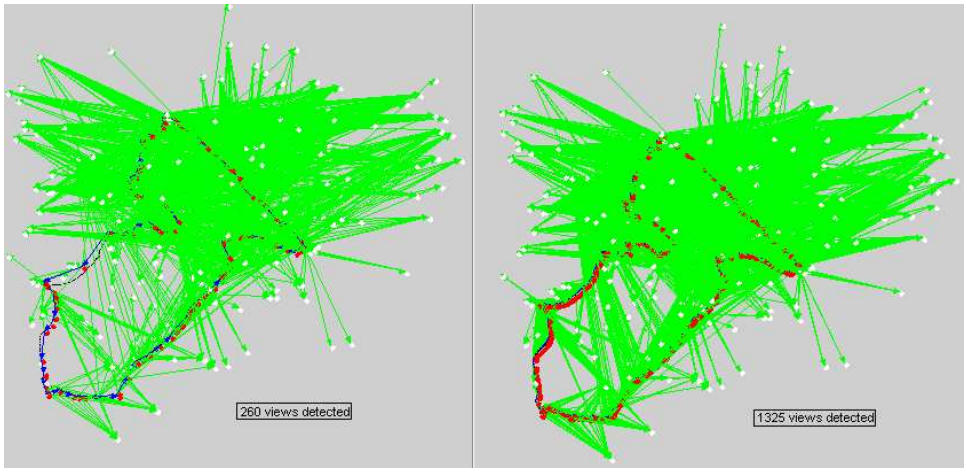


Abbildung 4.5: Der Agent begeht zwei Mal die gleiche Route, verwendet jedoch einen unterschiedlichen Schwellenwert für den Ähnlichkeitstest. In der Abbildung links ist der Schwellenwert null, rechts ist er eins. Die grünen Linien verbinden die gesehenen Objekte einer Ansicht mit deren Ursprung. Ist der Schwellenwert null, wird die topologische Anordnung der Objekte irrelevant. Jeder Ähnlichkeitswert liegt dann über dem Schwellenwert. Perspektive und 'source'-Objekt entscheiden dann allein, ob zwei Ansichten gleich sind. Es wird erwartet, dass nur wenige Ansichten gespeichert werden. Bei einem Schwellenwert von eins ist die Ansicht nur dann gleich, wenn neben 'source'-Objekt und Perspektive auch die topologische Anordnung der Objekte vollständig identisch ist. Damit gelangen die meisten Ansichten ins Langzeitgedächtnis. Betrachtet man die beiden Grafiken in der Abbildung, stellt man genau dieses Verhalten fest.

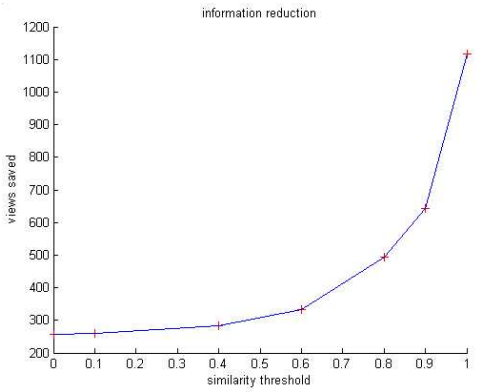


Abbildung 4.6: Die Abbildung zeigt wie sich die Anzahl gespeicherten Ansichten zu dem Ähnlichkeitsschwellenwert verhält. Das Langzeitgedächtnis war zuvor jeweils leer. Das Experiment wurde für die Schwellenwerte 0, 0.1, 0.4, 0.6, 0.8, 0.9 und 1.0 und für die Daten jeder der 10 Routen des Trainingssets aus Appendix A.1 wiederholt. Mittelt man die Ergebnisse der zehn Duchgänge, so erhält man die Kurve in der Abbildung. Die Anzahl gespeicherten Ansichten verhaltet sich exponentiell zum Schwellenwert.

Folglich wandern viele Ansichten ins Langzeitgedächtnis. In der Abbildung sind die visuellen Links der Ansichten grün eingezeichnet. Die Route im linken Teil der Abbildung mit einem Schwellenwert von 0.0 generiert weniger Ansichten, als die Grafik rechts mit einem Schwellenwert von 1.0.Das Resultat ist wie erwartet.
Das Experiment wurde für jede der Trainingsrouten in Appendix A.1, für die Schwellenwerte von 0, 0.1, 0.4, 0.6, 0.8, 0.9 und 1.0 je wiederholt. Die durchschnittliche Anzahl von generierten Ansichten ist in Abbildung 4.6 gegeben.

4.1.3 Asynchronität der Wahrnehmung

In Sektion 1.1 wurde gefordert, dass die Route, um von einem Ort A zu einem Ort B zu gelangen, ungleich der entgegengesetzten Route wahrgenommen wird, also von Ort B nach Ort A. Solche Wege werden als asynchron bezeichnet. In der mentalen Karte ist ein Weg durch aufeinanderfolgende 'Action Links' oder Routenobjekte definiert. Ein Routenobjekt ist eine Abfolge von 'Action Links' zwischen zwei Routenverzweigungen.

Ein 'Action Link' ist als eine Verbindung zwei aufeinanderfolgender Aktivitäten definiert. Ob eine Ansicht generiert wird, hängt davon ab, welche Objekte der Agent wahrnimmt. Ausschlaggebend ist die Blickrichtung des Agenten. Diese ist hier in der Simulation gleich wie die Bewegungsrichtung. Lässt man den Agenten beide Richtungen eines Weges ablaufen, generiert er unterschiedliche Ansichten. Voraussetzung ist, dass das Sichtfeld des Agenten kleiner als 360 Grad ist. Mit anderen Worten sind die 'Action Links' um von Ort A nach Ort B zu gelangen nicht dieselben, wie diejenigen, um von Ort B nach A zugelangen. Folglich sind auch die Routenobjekte nicht gleich.
Im Experiment wurde der Sachverhalt ausgetestet. In Abbildung 4.7 wird links ein Routenausschnitt gezeigt. Die roten Punkte geben die vorhandenen Routenverzweigungen an. Rechts in der Abbildung 4.7 ist der selbe Ausschnitt gegeben, mit den 'Action Links', die sich der Agent gemerkt hat. Es ist unschwer zu erkennen, dass die 'Action Links' auf dem Hin und Rückweg vollkommen verschieden sind. Die Orte an denen die Ansichten wahrgenommen werden unterschieden sich oft. Dies ist der Fall, wenn z.B. nur in die eine Richtung Objekte gesehen werden. Die Route auf dem Hin- und Rückweg sind also verschieden.

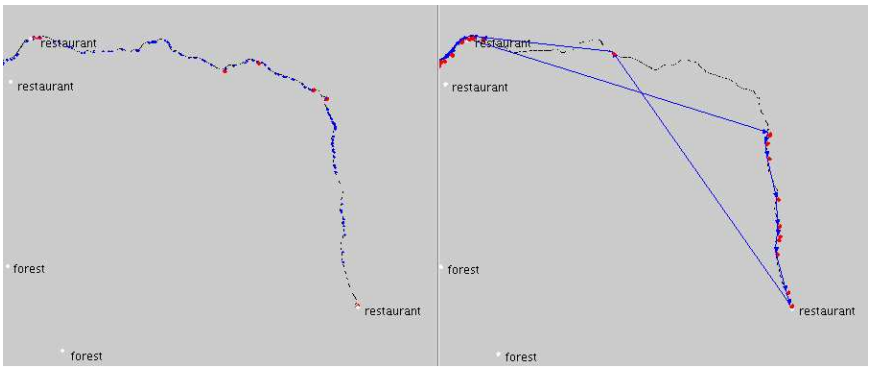


Abbildung 4.7: Aufgrund der unterschiedlichen Objekte, die der Agent wahrnimmt, wenn er sich in die eine oder andere Richtung auf einem Weg bewegt, wird er sich unterschiedliche Ansichten merken. Damit unterscheidet sich für den Agenten der Hin- vom Rückweg. In dem folgenden Ausschnitt aus der Ausführung einer Route ist der Effekt gut sichtbar. Die Grafik markiert die Koordinaten. an denen der Agent sich eine Ansicht merkt, rot und verbindet aufeinanderfolgende Ansichten mit einem blauen Pfeil. Der Ursache für die verschiedenen Ansichten liegt in der unterschiedlichen Blickrichtung des Agenten auf dem Hin- und Rückweg.

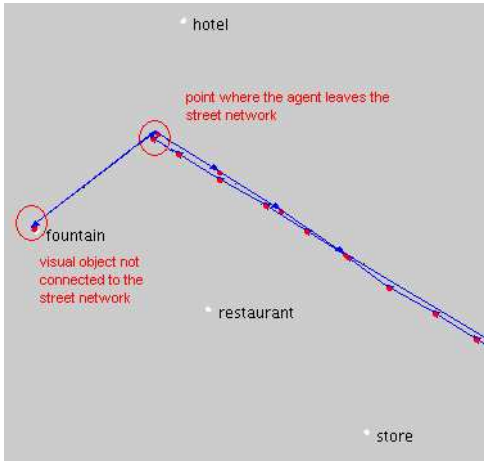


Abbildung 4.8: Die Abbildung zeigt, wie der Agent während der Ausführung einer Route das Strassennetzwerk verlässt und ein Objekt begeht, um dort eine Aktivität auszuführen. Das Objekt ist mit einem 'Action Link' verbunden. Der selbe Link definiert gerade auch ein Routenobjekt in der mentalen Karte. Auf dieser internen Route existieren keine Ansichten, da kein 'source'-Objekt vorhanden ist. Dem Routenobjekt liegt ja kein Strassenobjekt zugrunde. Diese umständliche Handhabung hat den Grund im Strassennetzwerk der Simulation, das eine zu kleine Auflösung hat. Begehbare Objekte, die nicht Strasse oder Verzweigung sind, sind nicht mit dem Strassennetzwerk verbunden. Um ein soches Objekt zu verlassen muss der Agent das Strassennetzwerk verlassen.

4.1.4 Begehbarkeit von Objekten

Es mag etwas banal tönen, die Begehbarkeit von Objekten auszutesten. Das Problem ist, dass visuelle Objekte, wie z.B. ein Haus, etc. in der Simulation nicht ans Strassennetzwerk angeschlossen sind. Dessen Auflösung ist zu klein. Besucht der Agent ein solches Objekt, wird er das Netzwerk am nächstgelegenen Knoten verlassen. Dies ist eine Annahme in der Simulation ALPSIM und stimmt nicht mit der Realität überein. Für den Weg von letzten Knoten des Strassennetzwerks zum gesehenen Objekt generiert der Agent intern ein Routenobjekt. In der Simulation ALPSIM verlässt der Agent das Netzwerk und wandert direkt zum Objekt.
In einem Experiment soll der Agent ein solches Objekt besuchen. In Abbildung 4.8 wird der entsprechende Routenausschnitt gezeigt. Erreicht der Agent das Objekt, beginnt er Ansichten auf sich selber zu generieren. Dies kann folgendermassen interpretiert werden. Der Agent ist im Restaurant und sieht das Restaurant von innen. Ein solche Ansicht wird immer erkannt. Das Routenobjekt der mentalen Karte ist definiert durch ein einzigen 'Action Link'. Dieser verbindet die letzte Ansicht, bevor das Strassennetzwerk verlassen wird und die Ansicht auf dem Objekt selbst. In der Abbildung ist der entsprechende 'Action Link' zu erkennen. Dasselbe gilt, wenn der Agent das Objekt wieder verlässt und auf das Strassennetzwerk zurückkehrt.

4.1.5 Routing basierend den Routenobjekten der mentalen Karte

Für die Planung der Aktivitätenroute werden die Routenobjekte der mentalen Karte verwendet. Ein Routenobjekt ist definiert als Verbindung zwischen zwei Routenverzweigungen. Genauso gut könnte man auch die 'Action Links' verwenden. Dies hätte zur Folge, dass interne Knoten kommuniziert werden müssten. Die Kommunikation müsste in der Simulation via Koordinaten geschehen. Das verursacht Probleme und zusätzlich Rechenaufwand bei der Wiedererkennung, da die Koordinaten variieren können. Diese Variation geschieht aufgrund von physikalischen Kräften, die den Agent auf einer Route beeinflus-

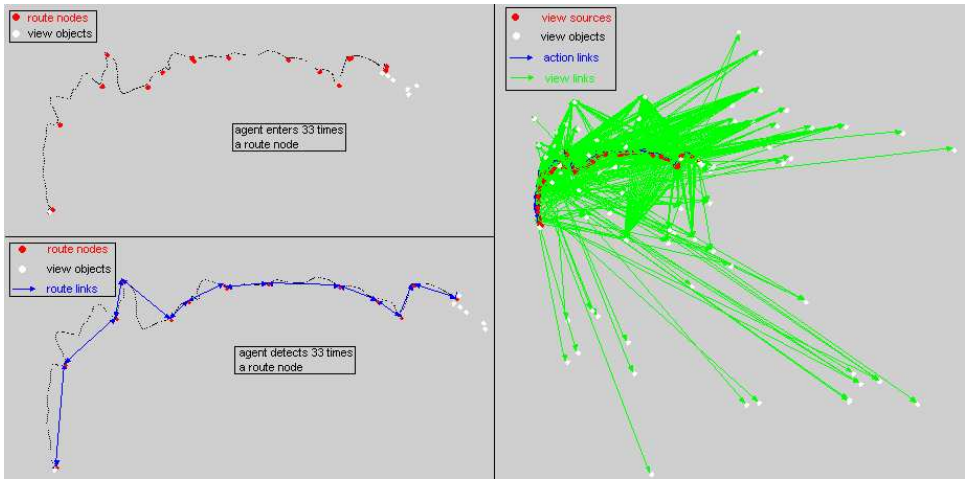


Abbildung 4.9: In der Grafik links oben ist die Route mit den dazugehörigen Verzweigungen gegeben. Der Agent läuft die Route, am rechten Ende beginnend, hin und zurück. Der Agent müsste 33 mal eine Verzweigung erkennen. Die Routenobjekte, die er erkennt sind unten links eingezeichnet. Alle 33 Verzweigungen werden detektiert. In der Grafik rechts sind die visuellen Links zu den gesehenen Objekten abgebildet. Die visuellen Objekte sind mit recht hoher Dichte und regelmässig verteilt. Der Agent kann von jeder Routenposition Objekte sehen.(Anmerkung: In den Grafiken links sind nur diejenigen visuellen Objekte eingezeichnet, die der Agent auch begehen kann)

sen. Es ist robuster, Knoten zu verwenden, die eindeutig definiert sind. Konkret sind damit Verzweigungsknoten des Strassennetzwerkes und visuell begehbare Objekte gemeint. Für eine Route sind ausschliesslich die Verzweigungsknoten entscheidend.

Eine Routenverzweigung wird erkannt, falls dort auch eine Ansicht erkannt wird. In der Simulation wird die Generierung von 'view-events' auf einer Verzweigung erzwungen. Der Agent hat auf einer Verzweigung eine höhere Sensitivtät, Objekte wahrzunehmen. Dies lässt sich damit begründen, dass man an einem Punkt der Entscheidung aufmerksamer ist. Eine erhöhte Sensitivität wird erreicht, indem der Bewertungszuwachs eines Objekts im Kurzzeitgedächtnis in Gleichung 2.3 über die zusätzliche Variable, 'decision pressure', beeinflusst wird. Der hier verwendete Wert ist 2.0.

Der Agent nimmt eine Ansicht auf einer Verzweigung mit erhöhter Wahrscheinlichkeit wahr. Dies nützt ihm jedoch nichts, falls es keine Objekte gibt, die er wahrnehmen kann. Der Agent ist dann blind und sieht die Verzweigung nicht. Genauso, falls der adaptive Schwellenwert im Kurzzeitgedächtnis zu hoch ist. Die Aufmerksamkeit liegt nur bei den Objekten deren Bewertung höher ist. Der adaptive Schwellenwert ist bestimmt durch die zuletzt wahrgenommen Objekte.

Es ist unsinnig, in einem Experiment zu zeigen, wie viele Routenverzweigungen prozentual in einer Route erkannt werden. Einzig die Verteilung und Dichte der visuellen Objekte im Raum ist ausschlaggebend. In zwei Beispielen soll dies verdeutlicht werden.

Abbildung 4.9 zeigt ein Beispiel, wo alle Routenobjekte erkannt werden. In der Grafik, links oben ist die Route mit den entsprechenden Routenverzweigungen gegeben. Der Agent wandert die Route hin und zurück. Er betritt dabei 33 mal eine Routenverzweigung. In der Grafik unten links sind die Routenobjekte eingezeichnet. Alle 33 Verzweigungen wurden wiedererkannt. In der Grafik rechts sind die visuellen Links der Ansichten eingezeichnet. Ein visueller Link ist die Verbindung von der momentanen Agentenposition zu einem gesehenen Objekt. Man erkennt, dass diese regelmässig und dicht aufeinander auftreten. Der Agent hat auf der ganzen Route die Möglichkeit, Objekte zu sehen.

Abbildung 4.10 zeigt ein Beispiel, wo nicht alle Routenobjekte erkannt werden. In der Grafik links oben ist wieder die Route mit den entsprechenden Routenverzweigungen ge-

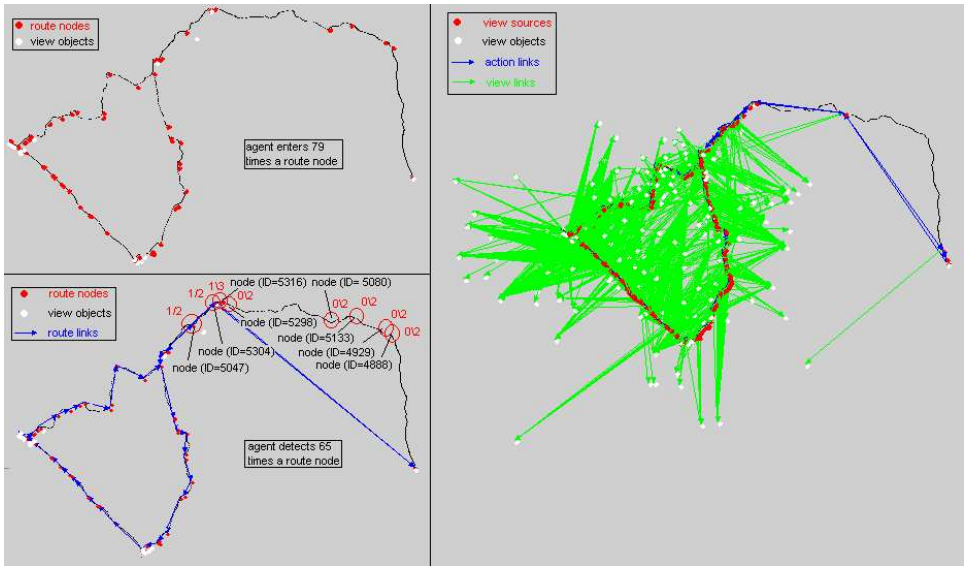


Abbildung 4.10: In der Grafik links oben ist die Route mit den dazugehörigen Verzweigungen gegeben. Der Agent sollte auf der Route 79 mal eine Verzweigung erkennen. Unten links sind die Routenobjekte eingezeichnet die der Agent erkennt. Er detektiert nur 65 mal eine Verzweigung. Die Verzweigungen, die nicht immer oder gar nie detektiert werden sind rot umkreist. Es ist angegeben, wie viele Male der Agent die Verzweigung besucht und wie viele Male er sie auch erkennt. Man betrachte nun die visuellen Links des Agenten in der Grafik rechts. Die wahrzunehmenden Objekte sind ungleichmässig in der Agentenwelt verteilt. Sieht der Agent keine Objekte in seiner Blickrichtung, kann er auch keine Verzweigung wahrnehmen. Wandert der Agent eine gewisse Zeit ohne Objekte wahrzunehmen, zerfallen die Bewertungen im Kurzzeitgedächtnis. Der Schwellenwert passt sich erst mit den Objekten, die danach wahrgenommen werden, an. Es ist eine gewisse Adaptionszeit nötig, um die neuen Objekte zu erkennen. In dieser Zeit kann der Agent auch keine Verzweigungen wahrnehmen. (Anmerkung: In den Grafiken links sind nur diejenigen visuellen Objekte eingezeichnet, die der Agent auch begehen kann)

location	viewable id	score	current threshold
$node(ID = 5298)$	–	–	–
$node(ID = 5298)$	–	–	–
$node(ID = 5316)$	–	–	–
$node(ID = 5080)$	–	–	–
$node(ID = 5133)$	–	–	–
$node(ID = 4929)$	–	–	–
$node(ID = 4888)$	–	–	–
$node(ID = 4888)$	–	–	–
$node(ID = 4929)$	–	–	–
$node(ID = 5133)$	–	–	–
$node(ID = 5080)$	–	–	–
$node(ID = 5316)$	$restaurant(ID = 5)$	0.50111	0.51964
$node(ID = 5304)$	–	–	–
$node(ID = 5047)$	–	–	–

Tabelle 4.2: Die Tabelle zeigt die Routenverzweigungen, die nicht erkannt wurden. 13 Mal ist der Grund, dass keine Objekte zu sehen waren. Einmal konnte zwar ein Objekt wahrgenommen werden, jedoch wurde der Schwellenwert im Kurzzeitgedächtnis nicht überschritten. Da zuvor über einen kurzen Zeitraum kaum Objekte wahrgenommen wurden, sind die Bewertungen der Objekte im Kurzzeitgedächtnis zerfallen. Der Schwellenwert muss zuerst wieder adaptieren. Im Fall von node (ID=5316) ist die Adaption bereits fortgeschritten, da bereits, nach der 'Blindphase' wieder Objekte wahrgenommen wurden.

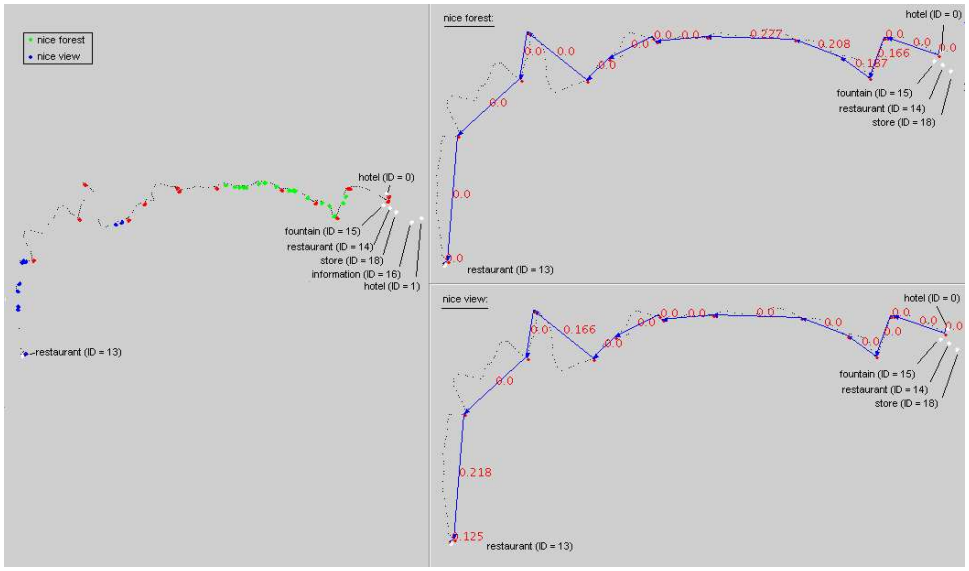


Abbildung 4.11: Die rechte Abbildung zeigt die Route, die der Agent ausführt. Die Orte an denen der Agent 'nice forest' Events wahrnimmt sind mit einem grünen Punkt markiert. 'Nice view' Orte sind blau markiert. Die Grafiken links zeigen die generierten Routenobjekte und die dazugehörigen Bewertungen. Die Grafik oben zeigt die Bewertung durch die 'nice forest' events verursacht, die Grafik unten diejenigen der 'nice view'. Die Bewertungsgrössen der Routen stimmen mit den Events auf der linken Grafik überein.

geben. Der Agent betritt 79 mal eine Routenverzweigung. In der Grafik unten links sind die entsprechenden Routenobjekte, die der Agent erkennt, eingezeichnet. Nur 65 mal erkennt er die Verzweigungen. Die Verzweigungen, die nicht oder nicht jedes Mal erkannt wurden, sind mit einem roten Kreis markiert. Es ist angegeben, wie viele Male sie erkannt wurde und wie viele Male der Agent sie benutzt hat. Man stellt fest, dass die nicht erkannten Routenverzweigungen recht konzentriert sind. Betrachtet man die Grafik rechts, wird auch klar, wieso: Bei den Verzweigungen, die nie erkannt werden fehlen die visuellen Objekte in der Blickrichtung des Agenten. Die Teilweise erkannten Verzweigungen können mehrere Ursachen haben. Der Agent hat jedes Mal, als er die Verzweigung betritt, eine andere Blickrichtung. Folglich sieht er auch unterschiedliche Objekte. Es kann durchaus sein, dass Objekte in die eine Richtung vorhanden sind, in der anderen jedoch nicht. Ist der Agent für eine bestimmte Zeit blind, zerfallen die Bewertungen im Kurzzeitgedächtnis. Der Schwellenwert bleibt jedoch unverändert. Es braucht eine gewisse Adaptionszeit des Schwellenwertes, bis er sich an die neuen Bewertungen angepasst hat. Dies hat zur Folge, dass visuelle Objekte nicht sofort wiedererkannt werden. In Tabelle 4.2 sind die Routenverzweigungen, die nicht erkannt wurden in korrekter Abfolge wiedergegeben. Nur einmal ist die Adaptionsverzögerung der Grund, dass die Verzweigung nicht erkannt wurde.

4.1.6 Bewertung von Routenobjekten

Der Agent wählt ein bestimmtes Routenobjekt nach seiner Bewertung. Die Bewertung eines Routenobjektes ist durch die 'feel-events' bestimmt. Ein 'feel-event' ist z.B. 'schöne Aussicht', 'schöner Wald' etc. . Die Häufigkeit, mit der ein solcher Eventtyp auf einem bestimmten Routenobjekt wahrgenommen wird, und die Präferenz des Agent für den Eventtyp, bestimmt die Grösse der Bewertung. Die Bewertung eines Routenobjektes ist zusammengesetzt aus den Bewertungen der einzelnen Eventtypen. Eine genaue Beschreibung ist in Sektion 2.4 gegeben.

In einem Experiment soll gezeigt werden, ob die erzeugte Bewertung eines Routenob-

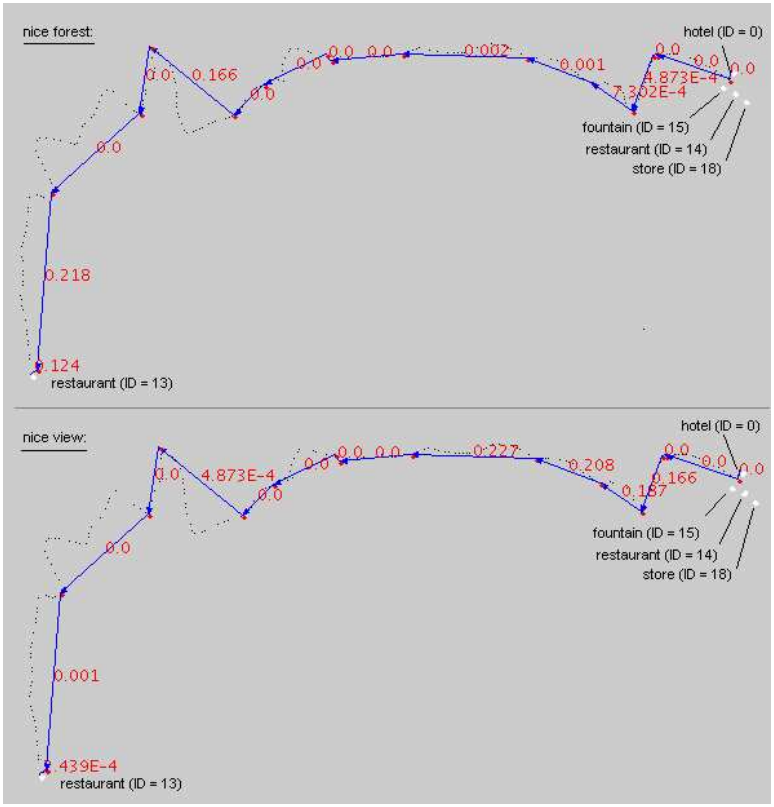


Abbildung 4.12: Die Ausgangslage ist die Route mit den Bewertungen aus Abbildung 4.11. Der Agent füht nun dieselbe Route mehrmals aus. Die 'nice view' Ereignisse werden nach dem ersten Durchgang mit den 'nice forest' vertauscht. Die Abbildung zeigt die Bewertungen nach zehn Durchgängen. Die 'nice forest' hat die initiale 'nice view' Bewertung adaptiert und umgekehrt.

jektes korrekt ist. In Abbildung 4.11 links ist die Route gegeben, die der Agent ausführt. Es werden hier 'feel-events' vom Typ 'nice forest' und 'nice view' gemessen. Sie sind an den genauen Koordinaten eingezeichnet, wo sie der Agent wahrnimmt. Der rechte Teil der Abbildung beinhaltet die generierten Routenobjekte der Strecke und jeweils die zugehörigen Bewertungen (rote Nummern) des entsprechenden Eventtyps. Man stellt fest, dass Wegstrecken auf denen in der linken Abbildung viele Ereignisse eines Eventtyps wahrgenommen wurden, rechts eine entsprechend hohe Bewertung haben.

Falls der Agent eine Route mehrmals begeht, wird er diese wiedererkennen. Die Bewertungen der alten Route werden aktualisiert, indem sie mit den neuen gemittelt werden. Ändert die Wahrnehmung über längere Zeit, wird der Agent allmählich, die neue Bewertung adaptieren. Der Agent muss dazu dieselbe Route mehrmals ablaufen. In Abbildung 4.12 wurden die 'nice forest' Ereignisse aus dem vorherigen Experiment, nach dem ersten Druchgang mit den 'nice view' Ereignisse vertauscht. Nach zehn Durchgängen haben die Bewertungen entsprechend adaptiert. Die 'nice forest' hat die initiale 'nice view' Bewertung adaptiert und umgekehrt.

Anmerkung: In Untersektion 2.4.1 wird die Bewertung eines Routenobjektes abhängig von der Tageszeit definiert. Die Attraktivitäten die von ALPSIM empfangen werden, sind zu jeder Tageszeit an einem Ort gleich. Wie in Untersektion 2.4.1 beschrieben, kann die Zeitabhängigkeit deshalb genauso als vernachlässigt betrachtet werden.

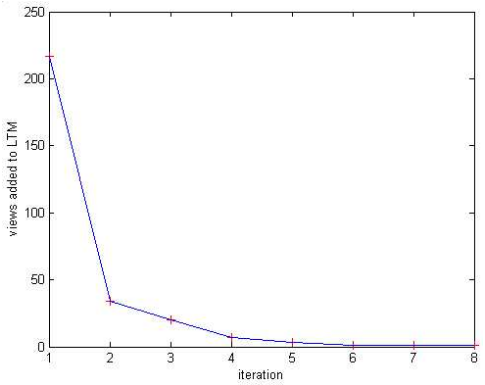


Abbildung 4.13: Im Experiment wandert der Agent dieselbe Route mehrmals ab. Die mentale Karte wurde vor dem ersten Durchgang neue Initialisiert. Die Grafik zeigt, die Anzahl Ansichten die der Agent in jedem Durchgang neu erkennt. Eine Ansicht ist neu, falls sie nicht schon im Langzeitgedächtnis vorhanden ist. Im ersten Durchgang werden 217 neue Ansichten erkannt, im zweiten noch 37, dann 20, 7, 3 ,1. Es tritt also schnell eine Konvergenz ein. Der Adaptive Schwellenwert des Kurzzeitgedächtnis ist beim ersten Durchgang null. In den darauffolgenden Durchgängen ist er durch die zuletzt wahrgenommenen Objekte des vorherigen Durchgangs bestimmt. Dies ist auch der Grund weshalb nicht nach dem ersten Durchgang schon alle Ansichten ins Langzeitgedächtnis gelangen. Bei einem fixen Schwellenwert sollte die Konvergenz bereits im zweiten Durchgang eintreffen.

4.1.7 Wiedererkennen von Informationen

Der Agent sollte imstande sein visuelle Ansichten wiederzuerkennen. Führt der Agent also eine Route zwei Mal aus, sollte er die Ansichten aus dem vorherigen Durchgang wiedererkennen.
In einem Experiment soll der Agent sein mentales Initialwissen aus einer einzigen Route aufbauen. Lässt man den Agenten danach immer wieder denselben Routenplan ausführen, ist zu erwarten, dass die Anzahl neu detektierten Routen schnell gegen null konvergiert. Aufgrund des adaptiven Schwellenwertes wird die Konvergenz nicht schon nach dem ersten Durchgang eintreffen. Der Grund ist folgender: Im ersten Durchgang beginnt der Agent mit einem Schwellenwert von null, da er noch nie ein Objekt wahrgenommen hat. Im den weiteren Durchgängen ist der Schwellenwert bestimmt durch die zuletzt wahrgenommenen Objekte im vorherigen Durchgang. Die Anzahl zusätzlich wahrgenommenen Objekte wird in jedem Durchgang kleiner werden. Der Schwellenwert konvergiert. Der Agent wird immer mehr die gleichen Objekte am Schluss eines Durchganges wahrnehmen. Es ist zu erwarten, dass schnell eine Konvergenz eintrifft. Abbildung 4.13 gibt an, wie viele neue Ansichten der Agent in jedem Durchgang detektiert anhand einer Beispielroute.
Nimmt man an der Agent begeht ein Routenstück mehrmals. Der adaptive Schwellenwert des Kurzzeitgedächtnisses kann zu beginn der Route jedes Mal anders sein. Dies ist davon abhängig, welche Objekte der Agent gerade vorher wahrgenommen hat. Je nach dem können so neue Ansichten etwas anders oder sogar neu erkannt werden. Genauso kann so eine Ansicht verloren gehen. Die Wiedererkennung einer Ansicht beruht auf einem Ähnlichkeitsprinzip (vgl. Sektion 2.2.2). Der Agent wird eine Ansicht deshalb höchstwahrscheinlich trotzdem wiedererkennen.

4.2 Selbständiges Erweitern des Wissens

Der Agent soll die gesammelten Informationen wiederverwenden können, um Aktivitätsplane mit räumlichem Wissen zu komplettieren. Ein Aktivitätsplan ist definiert als eine

abwechselnde Folge von punktuellen und räumlich ausgedehnten Aktivitäten. Ein Beispiel für eine punktuelle Aktivität ist 'essen'. Sie kann an einem bestimmten Ort ausgeführt werden. 'Wandern' hingegen hat eine räumliche Ausdehnung. Die Aktivität erstreckt sich über einen bestimmten Raum. Ein Aktivitätenplan startet und endet immer mit einer punktuellen Aktivität.
Der Algorithmus mit dem der Aktivitätenplan vervollständigt wird kann folgendermassen kurz umschrieben werden. Startend von Anfangsobjekt sucht der Agent mit Hilfe des Generalisationswissens (vgl. 2.13) alle Objekte, an denen er die nächste punktuelle Aktivität erfüllen kann. Mit Dijkstra oder A* versucht der Algorithmus einen Weg zu einem Kandidatenobjekt für die punktuelle Aktivität zu finden. Existiert kein solcher Weg oder überschreitet dieser ein bestimmtes maximales Zeitlimit, wird das Kandidatenobjekt gestrichen. Existiert ein gültiger Weg, wird zur nächsten punktuellen Aktivität rekursiv expandiert. Aktivitäten die nicht erfüllt werden können, werden übersprungen.
Eine genaue Beschreibung der Methoden findet sich in Kapitel 3. In den folgenden Untersektionen werden die in Kapitel 3 beschriebenen Eigenschaften anhand von Beispielen ausgetestet.

```
<request type="plan" id="1">
  <plan>
    <act name="hotel" id="0" endtime="0" />
    <act name="hike" duration="4000"/>
    <act name="eat" duration="1800"/>
    <act name="hike" duration="4000"/>
    <act name="hotel" id="0" starttime="9800"/>
  </plan>
</request>
```

Abbildung 4.14: Der Agent verwendet den in der Abbildung dargestellten Anfrageplan, um eine Route zu finden. Dazu benutzt er das räumliche Wissen in der mentalen Karte. Die Startzeit des Agenten, sowie das Startobjekt und Endobjekt sind gegeben. Der Agent soll vom Hotel mit der ID = 0 wandern gehen, nach gut 3 Stunden etwas essen und wieder ins Hotel zurückwandern. Die angegebenen Zeiten sind nur Vorschläge, an die sich der Agent so gut, wie es ihm das eingene Wissen erlaubt, halten soll.

4.2.1 Dijkstra vs. A*

Für das Finden eines Weges wird ein Dijkstra- oder A*-Algorithmus verwendet. Die mentale Karte kann dazu als Graph interpretiert werden. Die Routenverzweigungen sind Knoten. Eine Kante ist gegeben durch ein Routenobjekt. Für jede Kante ist eine Bewertung und eine Zeit gegeben. Für eine Route soll die Bewertung maximiert und die vorgegebene Zeit möglichst gut approximiert werden.
Dijkstra versucht die Bewertung zu maximieren, indem er jeweils die Kante mit maximaler Bewertung dazunimmt. Problematisch ist dabei die Zeit. Die Abweichung zur vorgegebenen Zeit der ganzen Route soll minimiert werden. Die bisher mit Dijkstra expandierte Route gibt jedoch keine passende Zeitabschätzung her. Diese berechnete Zeit entspricht nur derjenigen bis zum expandierten Knoten. Es ist jedoch eine Abschätzung für die ganze Route nötig. Nur so kann festgestellt werden wie viel man von der Vorgabe entfernt ist.
Dijkstra wird so abgeändert, dass er ein Routenobjekt nur dann dazunimmt, falls die maximal erlaubte Abweichung von der vorgegebenen Zeit nicht überschritten wird. Dabei entsteht folgendes Problem. Der Algorithmus nimmt eine neue bessere Kante in Kauf, auch wenn später das maximale Zeitlimit überschritten wird. Die abgebrochene Route besetzt dabei bestimmte Knoten. Andere Routen expandieren nicht mehr zu diesen Knoten, da sie eine geringere Bewertung haben. Die Route mit geringerer Bewertung hat unter umständen bisher weniger Zeit verbraucht und würde das Ziel erreichen. Das Problem wird in Abbildung 3.4 in Kapitel 3 beschrieben.

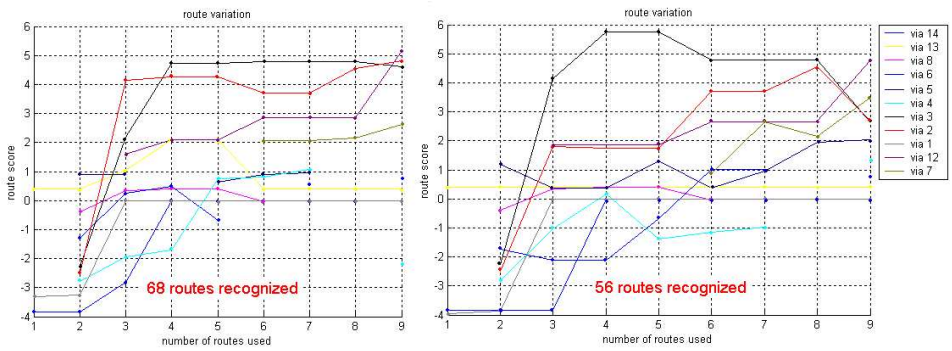


Abbildung 4.15: Im Experiment wird in jedem Durchgang das mentalen Wissen mit den Daten einer Trainingsroute erweitert. Auf der mentalen Karte wird eine passende Route zum jeweils gleichen Aktivitätenplan gesucht. Dieser startet und endet in einem vorgegebenen Objekt. Der Agent soll wandern und irgendwo etwas essen gehen. Die Routen sind charakterisiert durch das Objekt an dem sie die Aktivität 'essen' ausführen. Links in der Abbildung ist die Entwicklung der Dijkstra-Routen mit grösser werdendem mentalen Wissen gegeben. Rechts die A*-Routen. Eine Route verschwindet, falls die Aktivität 'essen' nicht innerhalb der maximalen zeitlichen Abweichung erreicht wird. Insgesamt findet Dijkstra 56 Routen, A* hingegen 68. Die Routen von A* tendieren weniger, das Zeitlimit zu überschreiten. Dies hat den Grund, dass A* eine Zeitabschätzung über die ganze Route beachtet und in die Bewertung miteinfliessen lässt.

A* versucht zusätzlich jeweils eine zeitliche Abschätzung für den noch unbekannten Teil der Route zu erhalten. Mit der Abweichung der Zeitabschätzung von der vorgegebenen Zeit wird die Bewertung eines Routenobjektes gewichtet. Die Bewertung verkleinert sich dabei proportional zur geschätzten Zeitabweichung. Ansonsten ist alles gleich wie bei Dijkstra. Um eine Zeitabschätzung zu erhalten, wird Dijkstra rückwärts ausgeführt, mit dem Zielobjekt als Start. A* verlangsamt sich gegenüber Dijkstra um einen Faktor der abhängig ist von der Anzahl Kandidatenobjekte für eine Aktivität. Für jedes Objekt muss Dijkstra einmal rückwärts, d.h. startend von Zielobjekt, ausgeführt werden. Bei gerichteten Kanten werden nun solche rückwärts bevorzugt.

In einem Experiment sollen die beiden Algorithmen gegen einander abgewogen werden. Dazu werden die Trainingsdaten aus Appendix A.1 benutzt. Iterativ wird die mentale Karte mit Hilfe der Daten einer Route aus dem Trainingsset erweitert. Auf jeder Kartengrösse wird jeweils der gleiche Aktivitätenplan ausgeführt. Es soll untersucht werden, wie sich die Bewertungen der Routen ändern mit dem zusätzlichen mentalen Wissen. Es wird erwartet, dass mit A* weniger Routen verloren gehen, da die Gesamtzeit der Route berücksichtigt wird.

Der Aktivitätenplan ist in Abbildung 4.14 gegeben. Die Route startet und endet im selben Hotel. Der Agent wandert, geht nach einer bestimmten Zeit etwas essen und wandert danach ins Hotel zurück. Durch das Verwenden von einer einzigen unbestimmten Aktivität lässt sich die Routenentwicklung gut mitverfolgen. Eine Route ist charakterisert durch das Objekt, wo die Aktivität 'essen' ausgeführt wird.

Die Bewertungen der Routen sind in Abbildung 4.15 gegeben. Links in der Abbildung sind die Routen abgebildet, die Dijkstra findet, rechts diejenigen von A*. Insgesamt findet Dijkstra 56 Routen, A* hingegen 68. Das Resultat ist wie erwartet. Die Bewertung einer Route fällt ab, wenn zusätzliches Wissen die Zeitabschätzungen zu einer optimalen Route verschlechtert. Je schlechter diese Abschätzung, desto falscher ist die Gewichtung der Bewertungen.

In Abbildung 4.16 ist links die Bewertung der besten Route in jeder Iteration angegeben. A* findet im Experiment die bessere Route. Am Anfang sind die beiden Algorithmen ungefähr gleich, nacher ist die Bewertung von A* ist meistens höher als bei den Dijkstra

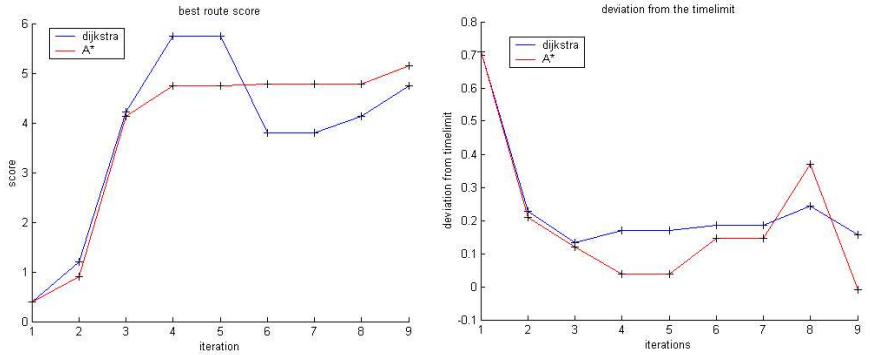


Abbildung 4.16: Links ist Bewertung der besten Routen aus jeder Iteration gegeben. Anfangs sind sie ungefähr gleich, nachher ist A* meistens besserer in der Bewertung. Rechts sind die Abweichungen der jeweils besten Route eines Durchganges gegeben. A* ist meistens näher am Zeitoptimum. Die Abweichung davon alleine ist nicht ausschlaggebend, wie gut eine Route ist. A* versucht nicht möglichst exakt ans Zeitlimit heranzukommen. Der Algorithmus versucht eine zeitliche Abweichung besser in Relation mit einem Bewertungszuwachs zu bringen.

Ergebnissen. Rechts sind die Gewichtungen der Bewertungen aufgrund der zeitlichen Abweichungen gegeben. Die Abweichung bezieht sich auf die ganze Route. Dazu werden die Zeitlimiten der einzelnen Aktivitäten zusammengezählt. Es kann vorkommen dass Dijkstra sich besser ans Zeitlimit annähert. A* versucht nicht möglichst exakt ans Zeitlimit heranzukommen. Der Algorithmus versucht eine zeitliche Abweichung besser in Relation mit einem Bewertungszuwachs zu bringen.
Für A* ist es wichtig, dass er an jedem Knoten eine Zeitabschätzung ins Ziel hat, die diese Relation optimal erfüllt. Die Zeitabschätzungen beruhen auf einem Dijkstra Algorithmus der am Zielobjekt startet. Dijkstra wählt eine Route innerhalb des gültigen Zeitkorridors aus. Die optimale Route ist die mit der besten Zeit-Bewertungs-Relation vom momentanen Knoten bis ins Ziel. Diese variiert, abhängig vom momentanen Knoten an dem sich der Agent befindet. Die Zeitabschätzung von Dijkstra muss nicht der optimalen Route entsprechen. A* ist ebenfalls nur eine Approximation an das optimale Ergebnis. Mit Dijkstra kann zufällig eine bessere Route gefunden werden. Falls A* keine Zeitabschätzung findet, wird nicht zum Knoten expandiert. Im Folgenden wird jeweils A* verwendet ausser es ist explizit anders angegeben.

4.2.2 Abhängigkeit des Plans von der Anzahl an Informationen

Es wird erwartet, dass sich eine Route mit zunehmendem Wissen ändert. Sie sollte sich dabei auch verbessern. Da approximative Algorithmen verwendet werden, wird nicht immer die beste Route gefunden. In Sektion 4.2.1 wurde gezeigt, dass sich eine Route bei grösser werdendem Wissen auch verschlechtern kann.
In Abbildung 4.17 und 4.18 wird gezeigt, wie sich eine Route ändert, falls auf unterschiedlich grossem mentalen Wissen gesucht wird. Die mentale Karte wurde jeweils aus den Daten der Trainingsrouten in Appendix A.1 konstruiert. Iterativ wurde jeweils der Reihe nach ein weiterer Datensatz dazugenommen, um die Karte zu vergrössern. Der Aktivitätenplan der zum Finden der Route benötigt wird ist in Abbildung 4.14 gegeben. Für das Routing zwischen zwei punktuellen Aktivitäten wird A* verwendet. Es sind nur die Iterationen angegeben, wo sich die Route deutlich ändert.

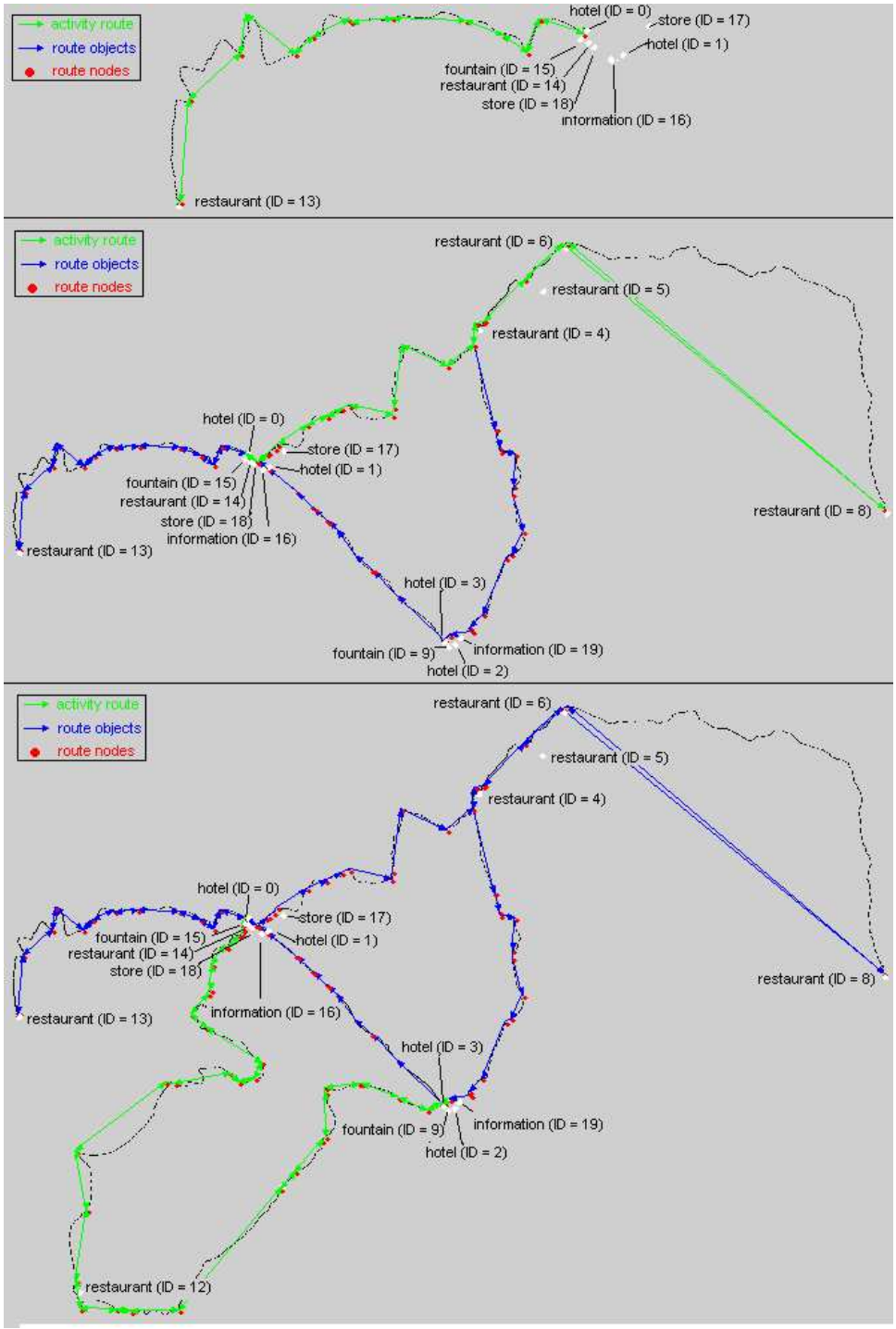


Abbildung 4.17: Der Agent vervollständigt den Aktivitätenplan mit räumlichem Wissen der mentalen Karte. Eine Route entsteht. An den verschiedenen Routenpositionen werden Aktivitäten ausgeführt. In der Abbildung wurde der gleiche Aktivitätenplan auf unterschiedlich grossem mentalen Wissen verwendet. Die grün gefärbten Routenobjekte gehören zur gefundenen Route. In der Abbildung sind drei unterschiedliche mentale Karten gegeben, geordnet nach der Grösse. Zwei weitere sind in Abbildung 4.18 gegeben. Man kann erkennen wie die Route mit zunehmendem Wissen variiert. Für das Routing zwischen zwei punktuellen Aktivitäten wurde hier A* verwendet.

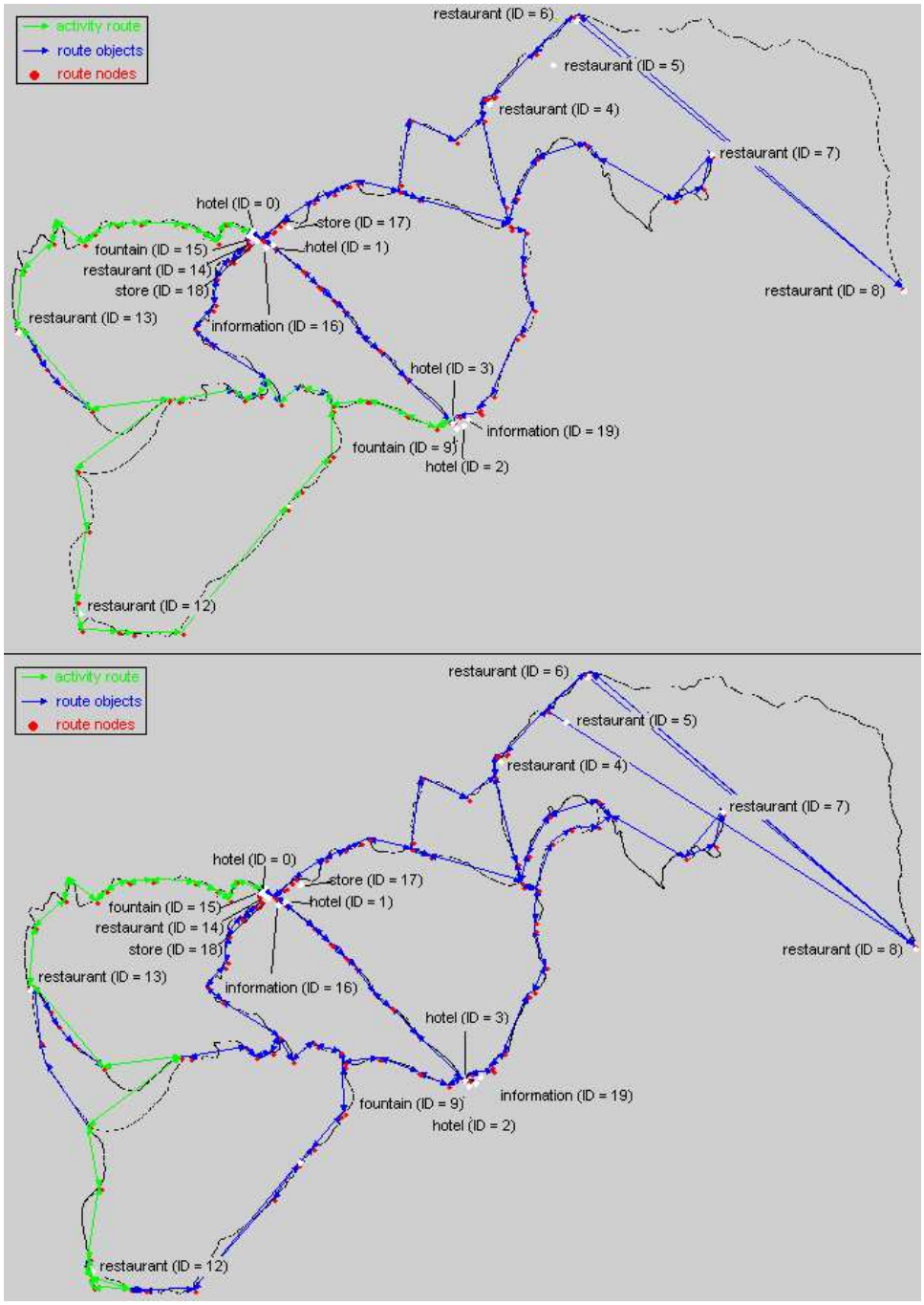


Abbildung 4.18: Der Agent vervollständigt den Aktivitätenplan mit räumlichem Wissen der mentalen Karte. Eine Route entsteht. An den verschiedenen Routenpositionen werden Aktivitäten ausgeführt. In der Abbildung wurde der gleiche Aktivitätenplan auf unterschiedlich grossem mentalen Wissen verwendet. Die grün gefärbten Routenobjekte gehören zur gefundenen Route. In der Abbildung sind zwei unterschiedliche mentale Karten gegeben, geordnet nach der Grösse. Drei weitere, kleinere Karten, sind in Abbildung 4.17 gegeben. Man kann deutlich erkennen wie die Route mit zunehmendem Wissen variiert. Für das Routing zwischen zwei punktuellen Aktivitäten wurde hier A* verwendet.

```
<request type="plan" agent="1">
        <plan>
                <act name="hotel" location_type="object_id" location="0" endtime="28800" />
                <act name="hike" duration="1800"/>
                <act name="sightseeing" duration="10800"/>
                <act name="hike" duration="1800"/>
                <act name="hotel" location_type="object_id" location="3" starttime="42200"/>
        </plan>
</request>
```

Abbildung 4.19: Der Aktivitätenplan beruht auf zwei unabhängigen Teilen in der mentalen Karte. Diese sind durch ein Objekt verbunden, dass von beiden Teilen visuell wahrgenommen wird. Das Objekt ist vom Typ 'Schloss' (castle). Der Agent started im Objekt 'Hotel' mit der ID = 0 und soll eine Sehenswürdikeit besuchen gehen. Danach muss der Agent in das Objekt 'Hotel' mit der ID = 3 zurückkehren. Für den Weg sind je eine halbe Stunde eingeplant.

4.2.3 Verwenden von visuellen Daten in der Planung

Der Agent verwendet zur Vervollständigung eines Aktivitätenplans zwei Typen von Wissen aus der mentalen Karte. Einerseits benutzt er Objekte, die er direkt erfahren hat, d.h. die er schon einmal besucht hat, anderseits solche, die er nur visuell wahrgenommen hat. Ein visueller Link zu einem Objekt kann nur einmal benutzt werden. Bei der Ausführung des Aktivitätenplans vervollständigt die Simulation die fehlenden Routeninformationen des Aktivitätenplans. Dies kann als konsultieren z.B. einer Wanderkarte interpretiert werden. Aufgrund der bei der Ausführung neu wahrgenommenen Informationen wird der Agent in Zukunft die genaue Route kennen.
In einem Experiment soll gezeigt werden, dass ein solcher visueller Link das Wissen des Agenten expandiert. Die initiale mentale Karte wird mit Hilfe der Daten von zwei Routen aufgebaut. Die Routen überschneiden sich nicht. Sie sind jedoch durch einen visuellen Link zu einem begehbaren Objekt verbunden. Das Objekt hat den Typ 'Schloss' (castle). Das Schloss kann also von beiden Teilrouten wahrgenommen werden. Die Routen stammen aus den Trainingsset in Appendix A.1.
In Abbildung 4.20 ist in der obersten Grafik, das mentale Wissen auf der Ebene der 'Action Links' gegeben. Die visuellen Links zu den begehbaren Objekten sind grün eigezeichnet. Die restlichen visuellen Links wurden der Übersicht halber weggelassen. In der zweiten Grafik ist die mentale Karte auf der Ebene der Routenobjekte gegeben. Es wird ersichtlich, dass das Objekt vom Typ 'Schloss' nur visuell wahrgenommen wird.
Der Agent führt nun den Plan aus Abbildung 4.19 aus. Dabei soll er vom Hotel mit ID=1 startend nach Hotel mit ID=3 wandern. Auf dem Weg möchte er eine Sehenswürdigkeit besuchen. Im Generalisationswissen ist der Typ 'castle' als einziger Untertyp der Aktivität 'sightseeing' definiert. Es existiert nur ein Objekt vom Typ Schloss.
In der untersten Grafik in Abbildung 4.20 ist die Route, die der Agent passend zum Aktivitätenplan findet angegeben. Der Agent startet wie vermutet im Hotel mit der ID=1. Er benutzt einen visuellen Link von dem oberen mentalen Kartenteil ins Schloss. Ein zweiter visueller Link bringt ihn in die untere Teilkarte. Von dort geht er weiter ins Hotel mit der ID=3. Der genaue Plan in XML Form ist in Abbildung 4.21 gegeben.
Der Aktivitätenplan wir vom Agenten ausgeführt. Von der Simulation erhält er die entsprechenden Events zurück. In Abbildung 4.22 oben, ist die aktuelle mentale Karte gegeben. Man erkennt, wie diese mit einem Weg via das Objekt vom Typ 'Schloss' ergänzt wurde. Die mentale Karte hat expandiert. Der Grund dafür ist der verwendete visuelle Link. Lässt man den Agenten den Aktivitätenplan aus Abbildung 4.19 erneut vervollständigen, findet er die Route in Abbildung 4.22 unten. Die Route benutzt das dazugewonnene mentale Wissen. In Zukunft wird der Agent diese Route verwenden.

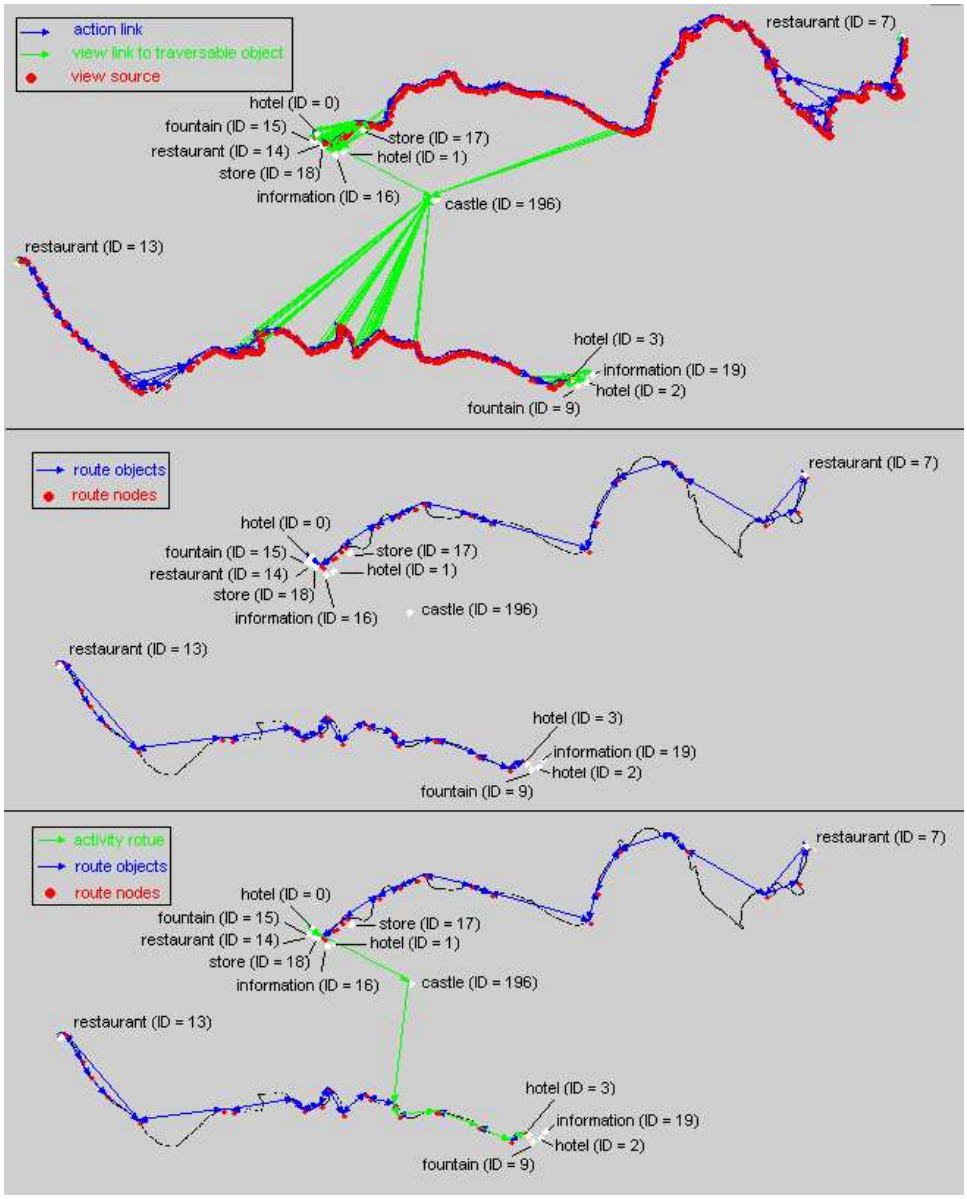


Abbildung 4.20: Im Experiment wird die mentale Karte des Agenten mit den Daten zweier Routen initialisert. Betrachtet man die oberen zwei Grafiken stellt man fest, dass die mentale Karte sich in zwei Teilstücke aufteilen lässt. Diese sind nur durch visuell wahrgenommene Objekte verbunden. Mit anderen Worten es existiert keine direkte Verbindung. Eines dieser Objekte von Typ 'Schloss' (castle), mit der ID = 196 ist vom Agenten begehbar. Führt der Agent den Plan aus Abbildung 4.19 aus, wird er gezwungen dieses zu verwenden, da das Start und Zielobjekt je auf einem anderen Teilstück der mentalen Karte liegt. Die gefundene Route ist in der untersten Grafik grün eingefährbt. Sie beinhaltet zwei visuelle Links

4.2.4 Einfluss der individuellen Präferenzen des Agenten auf den Aktivitätenplan

Jeder Agent hat individuelle Präferenzen. Diese beeinflussen die Bewertung der Informationen in der mentalen Karte. Es werden zwei Typen von Informationen bewertet. Einerseits werden die Basisobjekte im Kurzzeitgedächtnis bewertet, anderseits die Routenobjekte in

```
<request type="plan" agent="1">
    <plan agent="1" score="-2.5134263534380064" >
        <act name="hotel" location_type="object_id" location="0" end_time="28800" />

        <act name="hike_start" location_type="object_id" location="0" expected_time="893" start_time="28800" />
        <act name="hike_waypoint" location_type="node_id" location="4086" />
        <act name="hike_waypoint" location_type="node_id" location="4058" />
        <act name="hike_end" location_type="object_id" location="196" end_time="29693" />

        <act name="sightseeing" location_type="object_id" location="196" start_time="29693" end_time="40493" />

        <act name="hike_start" location_type="object_id" location="196" expected_time="1953" start_time="40493" />
        <act name="hike_waypoint" location_type="node_id" location="3283" />
        <act name="hike_waypoint" location_type="node_id" location="3237" />
        <act name="hike_waypoint" location_type="node_id" location="3249" />
        <act name="hike_waypoint" location_type="node_id" location="3250" />
        <act name="hike_waypoint" location_type="node_id" location="3201" />
        <act name="hike_waypoint" location_type="node_id" location="3196" />
        <act name="hike_waypoint" location_type="node_id" location="3135" />
        <act name="hike_waypoint" location_type="node_id" location="3167" />
        <act name="hike_end" location_type="object_id" location="3" end_time="42446" />

        <act name="hotel" location_type="object_id" location="3" start_time="42446" />
    </plan>
</request>
```

Abbildung 4.21: Die Abbildung zeigt die Antwort des Agenten auf die Anfrage in Abbildung 4.19. Die zugehörige mentale Karte, sowie der graphische Lösung ist in Abbildung 4.20 gegeben. Der Plan wie er abgebildet ist wird an die Simulation weitergeschickt, wo ihn der Agent ausführt.

der mentalen Karte. Letztere setzt sich zusammen aus verschiedenen Attraktivitätstypen, wie ’schöne Aussicht’ oder ’schöner Wald’ deren Vorkommen auf dem entsprechenden Routenobjekt wahrgenommen werden. Wir nehmen hier an, dass sich die Bewertungen der verschiedenen Attraktivitätstypen nicht gegenseitig beeinflussen. Eine genaue Beschreibung der Attraktivität ist in Sektion 2.4) gegeben.

Der Aktivitätenplan setzt sich aus einer Abfolge von punktuellen und räumlich ausgedehnten Aktivitäten zusammen. Eine räumlich ausgedehnte Aktivität beschreibt dabei meistens die Art, wie ein Objekt erreicht werden kann, um dort eine punktuelle Aktivität auszuführen. Dabei wird immer ein bestimmter Weg im Raum zurückgelegt. Der Weg ist gegeben durch verschiedene Routenobjekte. Der Agent wählt diese aufgrund deren Bewertung und einer Zeitlimite für die Aktivität. Die Präferenz für einen Attraktivitätstypen beeinflusst die Bewertung und somit auch die Routenwahl.

In einem Experiment soll dieser Einfluss deutlich gemacht werden. Das verwendete mentale Wissen beinhaltet die Daten der Trainingsrouten in Appendix A.1. Der Agent nimmt zwei verschiedene Attraktivitätstypen wahr: ’schöne Aussicht’ (nice view) und ’schöner Wald’ (nice forest). Der zu komplettierende Aktivitätenplan ist in Abbildung 4.23 gegeben. Abbildung 4.24 zeigt die Verteilung der einzelnen wahrgenommenen Attraktivitäten. Auf den Strecken, wo z.B. viel ’schöner Wald’ wahrgenommen wird ist die Bewertung der Routenobjekte entsprechend hoch (vgl. auch Abbildung 4.11). Eine Abbildung mit den direkten Routenbewertungen wurde weggelassen. Aufgrund der hohen Anzahl von Routenobjekten ist es nicht möglich, die Objekte mit den zugehörigen Bewertungen leserlich darzustellen.

In einem ersten Durchgang hat der Agent eine Präferenz für ’schöner Wald’ von 1.0 und für ’schöne Aussicht’ von 0.0. Der Wert 1.0 steht dafür, dass der Agent den Attraktivitätstyp mag. Eine Präferenz von 0.0 bedeutet, dass es dem Agent gleichgültig ist, ob er die Route besucht oder nicht. In Abbildung 4.25 oben ist die Route gezeigt, die der Agent für den Aktivitätenplan findet. In einem zweiten Durchgang wird die Präferenz von ’schöner Wald’ von 1.0 auf 0.0 geändert und diejenige von ’schöne Aussicht’ von 0.0 auf 1.0. Das Resultat für die zum Attraktivitätenplan passende Route ist in Abbildung 4.25 unten gegeben. Im ersten Durchgang hat der Agent also eine Vorliebe für ’schöner Wald’ im zweiten eine Vorliebe für ’schöne Aussicht’. Es ist leicht zu erkennen, dass die gefundenen Routen verschieden sind. Vergleicht man Route eins mit dem Vorkommen von ’schöner Wald’ in Abbildung 4.24, stellt man fest, dass die Route genau die entsprechenden Wege beinhalten. Das gleiche gilt in der zweite Route für ’schöne Aussicht’. Die Route stimmt mit dem

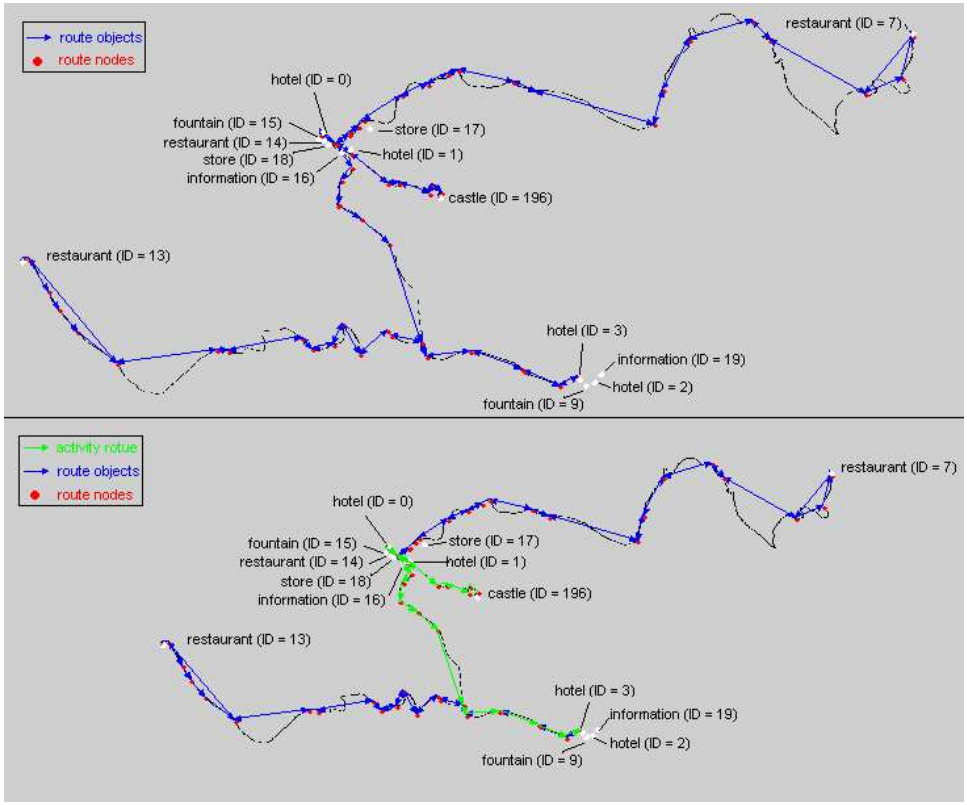


Abbildung 4.22: In der oberen Grafik ist das mentale Wissen nach dem Ausführen vom Plan aus Abbildung 4.21 abgebildet. Vergleicht man die Karte mit der bisherigen in Abbildung 4.20, so stellt man fest, dass zusätzlich eine Route via das Objekt vom Typ 'Schloss' und der ID = 196 die beiden mentalen Kartenstücke direkt verbindet. Das mentale Wissen hat expandiert. Sucht man erneut eine Route für denselben Aktivitätenplan wie vorhin (vgl. Abbildung 4.19), wird das neue Wissen verwendet. Die neue Routen ist in der unteren Grafik gegeben.

```
<request type="plan" agent="1">
        <plan>
                <act name="hotel" location_type="object_id" location="0" endtime="28800" />
                <act name="hike" duration="3600"/>
                <act name="eat" duration="3600"/>
                <act name="hike" duration="5400"/>
                <act name="hotel" location_type="object_id" location="0" starttime="41400"/>
        </plan>
</request>
```

Abbildung 4.23: Der Attraktivitätsplan verlangt vom Agenten eine Route zu finden, die im Objekt 'Hotel' mit der ID = 0 startet und endet. Der Agent soll nach einer Stunde wandern etwas essen gehen und dann ins Hotel zurückkehren.

Vorkommen von 'schöne Aussicht' überein.
Wie erwartet beeinflusst die Präferenz für einen bestimmten Attraktivitätstyp die Routenwahl, um einen Attraktivitätsplan zu vervollständigen.

4.2.5 Ändern des Akivitätenplans

In den bisherigen Experimenten waren die verwendeten Aktivitätenanfragepläne jeweils nach einem sehr strickten Muster aufgebaut. Es ist jedoch nicht immer möglich, dass der

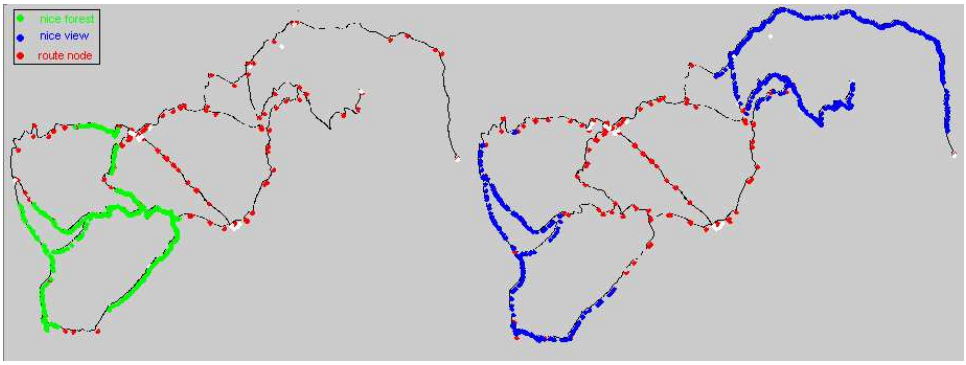


Abbildung 4.24: Für das Initialisieren der mentalen Karte wurden die Daten der Trainingsrouten in Appendix A.1 verwendet. Die Abbildung zeigt die Karte. Die Vorkommen der einzelnen Attraktivitätstypen sind farbig eingezeichnet. Ein grüner Punkt steht für einen Ort an dem schöner Wald wahrgenommen wurde, ein blauer Punkt für einen Ort mit schöner Aussicht. Die roten Punkte markieren die Verzweigungen des Strassennetzwerks. Auf einem Wegstück mit einem gehäuften Vorkommen eines bestimmten Attrakivtätstyps, werden die Routenobjekte entsprechend hoch Bewertet.

Agent eine Aktivität erfüllen kann. Es kann sein, dass er kein Objekt kennt, welches für die Ausführung der Aktivität geeignet wäre. Ein anderer Grund ist, falls kein entsprechendes Objekt innerhalb der maximalen Zeitabweichung existiert . Trotzdem sollen die restlichen Aktivitäten ausgeführt werden. Die oberste Priorität für den Agenten ist es die Zielaktivität zu erreichen. Damit dies möglich ist, wird er Aktivitäten, die er nicht erfüllen kann auslassen.

In einem Experiment soll die Situation ausgetestet werden. Die mentale Karte entspricht derjenigen in Abbildung 4.26 links. Der verwendete Aktivitätenplan ist in Abbildung 4.27 links gegeben. Der Agent soll startend vom Objekt vom Typ 'Hotel' und der Identität 2 Proviant kaufen, sich an einer Information informieren und dann wieder ins Hotel zurück gehen. Um Proviant zu kaufen ist ein Objekt vom Typ 'Store' nötig. In Abbildung 4.26 links ist ersichtlich, dass es kein solches Objekt in der näheren Umbegung vom Startort gibt. Im vervollständigten Plan des Agenten in der Abbildung 4.27 rechts, wurde die Aktivität übersprungen. Kann der Agent keine Aktivität ausführen bleibt er wo er ist. Eine andere Möglichkeit wäre einfach einen neuen Aktivitätenplan zu generieren und die neuen Aktivitäten auszuprobieren versuchen.

Der Planungsalgorithmus ist so ausgelegt, dass der Agent theoretisch auch zwei Aktivitäten am gleichen Ort ausführen kann. Dazu müssen zwei punktuelle Aktivitäten aufeinanderfolgen. Kann der Agent die Aktivitäten nicht am selben Ort ausführen, schaltet er eine Aktivität 'wandern' dazwischen. Die Wanderzeit wird von den zwei punktuellen Aktivitäten je zur Hälfte abgezogen.

In einem zweiten Experiment soll die entsprechende Eigenschaft getestet werden. Die benutzte mentale Karte ist in Abbildung 4.26 rechts gegeben. Für das Experiment wurde der Typ vom Objekt mit der Identität 18 von 'store' auf 'shopping_center' geändert. Der Agent soll versuchen mehrere Aktivitäten dort, d.h. am selben Ort auszuführen. Dies ist nur möglich, wenn im Generalisationswissen beide Aktivitäten Supertyp von dem Typ des Objektes sind, wo sie ausgeführt werden sollen.

Im Moment wird via Generalisationswissen gesteuert, ob an einem Objekt eine Aktivität ausgführen kann. Die Idee ist in Zukunft die Ausführbarkeit zusätzlich von der Tageszeit abhängig zu machen. Dies ist im Moment noch nicht implementiert.

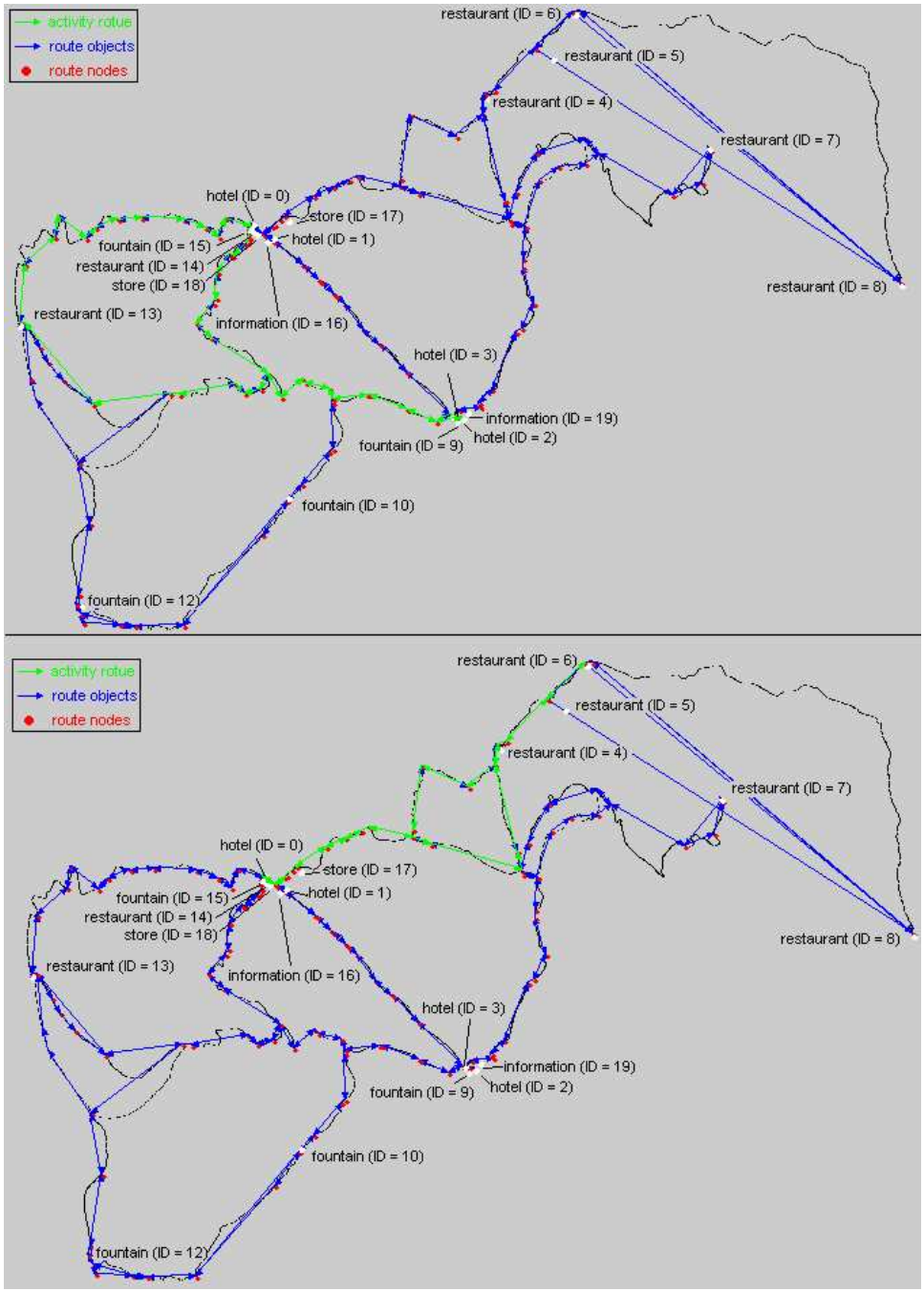


Abbildung 4.25: Links in der Abbildung ist die Route, die der Agent findet mit einer Vorliebe für 'schöner Wald'. Die schöne Aussicht ist ihm dabei unwichtig. Vergleicht man die Route mit dem Vorkommen der Wahrnehmungen für einen bestimmten Attraktivitätstyp, stellt man fest, dass sich die Route gemäss Präferenzen daran angepasst hat. Rechts in der Abbildung ist die Route, die der Agent mit einer Vorliebe für 'schöne Aussicht' findet. Wieder entspricht die Route Orten, wo zuvor schöne Aussicht wahrgenommen wurde. Die Route ist Abhänging von dem Vorkommen der einzelnen Attraktivitätstypen und den Präferenzen des Agenten für den einzelnen Typ.

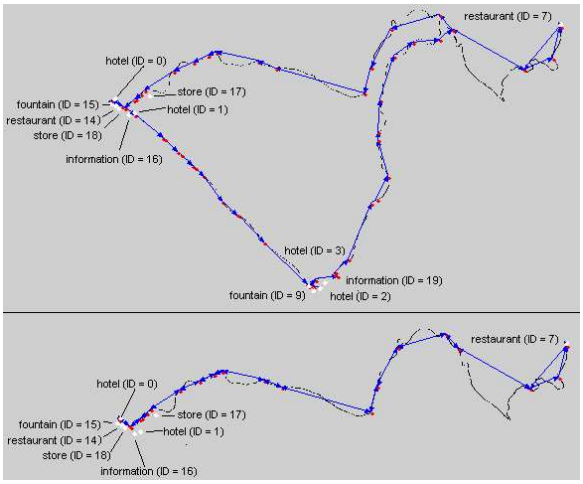


Abbildung 4.26: Die Grafik oben zeigt das mentale Wissen, das im ersten Experiment zur Verfügung steht. Unten ist die mentale Karte des zweiten Experiments gegeben.

```
<request type="plan" agent="1">
  <plan>
    <act name="hotel" location_type="object_id" location="2" endtime="0" />
    <act name="hike" duration="600" />
    <act name="buy_provisions" duration="900"/>
    <act name="hike" duration="600" />
    <act name="get_local_information" duration="300"/>
    <act name="hike" duration="600" />
    <act name="hotel" location_type="object_id" location="2" starttime="3000" />
  </plan>
</request>
```

```
<plan agentID="1" score="-1.9668941602175567" >

  <act name="hotel" location_type="object_id" location="2" expected_time="0" />

  <act name="hike_start" location_type="object_id" location="2" expected_time="419" />
  <act name="hike_waypoint" location_type="node_id" location="3152" />
  <act name="hike_waypoint" location_type="node_id" location="3163" />
  <act name="hike_waypoint" location_type="node_id" location="3190" />
  <act name="hike_waypoint" location_type="node_id" location="3217" />
  <act name="hike_end" location_type="object_id" location="19" />
  <act name="get_local_information" location_type="object_id" location="19" expected_time="300" />

  <act name="hike_start" location_type="object_id" location="19" expected_time="181" />

  <act name="hike_waypoint" location_type="node_id" location="3190" />
  <act name="hike_waypoint" location_type="node_id" location="3163" />
  <act name="hike_waypoint" location_type="node_id" location="3152" />
  <act name="hike_end" location_type="object_id" location="2" />

  <act name="hotel" location_type="object_id" location="2" expected_time="0" />
</plan>
```

Abbildung 4.27: Links in der Abbildung ist der verwendete Anfrageplan an die mentale Karte gegeben. Der Agent soll von dem Objekt vom Typ 'Hotel' mit der Identität 2 startend, Proviant kaufen gehen und danach sich an der Information informieren gehen. Rechts in der Abbildung ist die Antwort des Planungsalgorithmus gegeben. Die Aktivität 'Proviant einkaufen' wurde übersprungen. Dies ist nicht verwunderlich. In der lokalen Umgebung des Agenten ist keine Einkaufsmöglichkeit, die das Zeitlimit erfüllt vorhanden. (Einkaufsmöglichkeit entspricht einem Objekt vom Typ 'store').

4.2.6 Dynamische beinflussbare Attraktivitäten

In Untersektion 2.4.2 und Sektion 3.6 werden Bewertungen besprochen, die eine gegenseitige Abhängigkeit haben. Wird die Bewertung auf einem Routenobjekt nicht wie erwartet wahrgenommen, so vermutet der Agent, dass die Bewertung der ganzen Route nicht mehr stimmt. Er probiert sein Verhalten entsprechend anzupassen, indem er eine neue Route sucht, die den Gegebenheiten angepasst ist. Mit anderen Worten, versucht der Agent eine neue, an die neuen Bewertungen angepasste Route zu finden für die Aktivitäten, die er noch nicht ausgeführt hat.
Die Fähigkeit soll in einem Experiment getestet werden. Der Agent kann dabei die Attraktivitäten: 'Sonnenschein', 'schöner Wald' und schöne Aussicht wahrnehmen. In Abbildung 4.30 oben, sind die Objekte, die die bei der Initialisierung der mentalen Karte wahrgenommen werden, in einem Koordinatensystem dargestellt. Die farbigen Punkte stellen die verschiedenen Attraktivitästypen dar. Ein Ort, wo der Agent eine entsprechende Attraktivität wahrnimmt, ist durch einen entsprechenden Punkt markiert. Aufgrund der Präferenzen des Agenten für die einzelnen Attraktivitätstypen und dem vorgegebenen Attraktivitätsplan

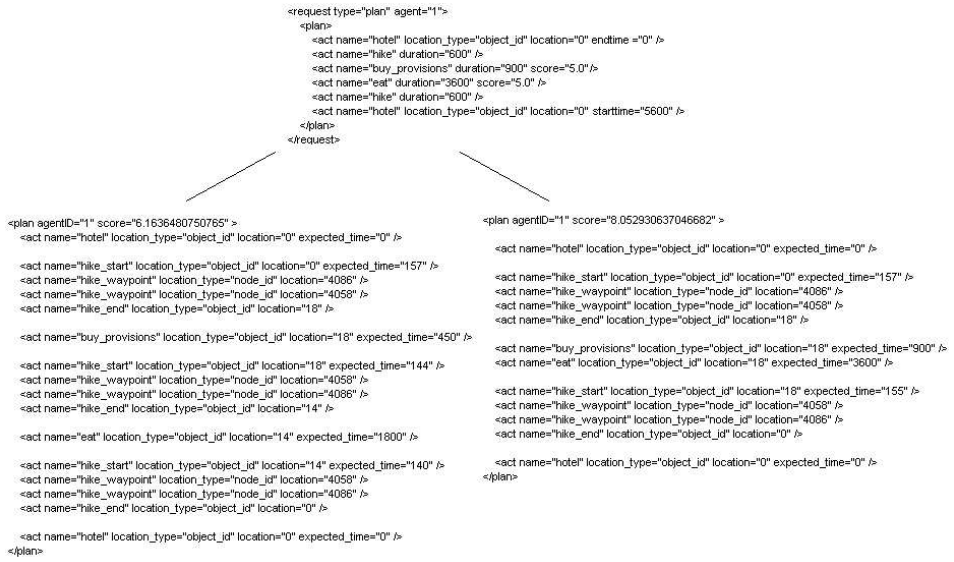


Abbildung 4.28: Oben ist der Anfrageplan an die Simulation abgebildet. Unten links ist das Resultat, das uns der verwendete Algorithmus zurückliefert, falls im Generalisationswissen beide Aktivitäten einem bestimmten Objekttyp zugewiesen werden können. Beide Aktivitäten werden am selben Objekt ausgeführt. Unten rechts ist das Resultat, das uns der verwendete Algorithmus zurückliefert, falls die beiden Aktivitäten nicht im demselben Objekt ausgeführt werden können. Der Algorithmus fügt eine Aktivität ein, die es dem Agenten erlaubt, ein anderes Objekt zu suchen. Die Aktivitäten werden im resultierenden Plan an verschiedenen Orten ausgeführt.

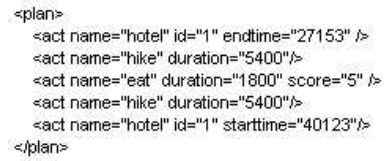


Abbildung 4.29: Der Aktivitätenplan beginnt bei einem Objekt vom Typ 'Hotel' und der Identität 1. Nach etwa anderthalb Stunden wandern soll der Agent etwas essen gehen. Danach soll er nochmals anderthalb Stunden wandern und schliesslich wieder ins selbe Hotel zurückzukehren.

4.29 entscheidet sich der Agent für die in Abbildung 4.30 unten grün markierte Route. (Sämtliche Präferenzen sind 0.5).
Bei der Ausführung des Aktivitätenplans erwartet der Agent die selben Attraktivitäten wahrzunehmen wie in Abbildung 4.30 oben dargestellt. Entgegen den Erwartungen des Agenten soll dieser jedoch die Objekte wie sie in Abbildung 4.32 oben angegeben sind wahrnehmen. Der Agent empfindet ab einem bestimmten Zeitpunkt entgegen seinen Erwartungen kein 'Sonnenschein' mehr. Es wird erwartet, dass sobald der Agent die Abweichung bemerkt, die Auswirkungen auf die Bewertung der Teilrouten die im restlichen Plan prüft. Dazu berechnet er die Bewertungen der restlichen Routenobjekte mit Einbezug der erwarteten Abweichung neu. Falls diese Abweichung von der bisherigen Vermutung einen definierten Schwellenwert überschreitet, geht er zur nächsten Routenverzweigung im bisherigen Plan und beginnt mit der Neuplanung der restlichen Route. Der Agent muss in der Lage sein die Ausführung seines Aktivitätenplan jeweils mitzuverfolgen. Der neue Plan ist dann den erwarteten Veränderungen entsprechend angepasst.
Der Agent, soll die initiale mentale Karte aufbauen und die gefundene Route zum Akti-

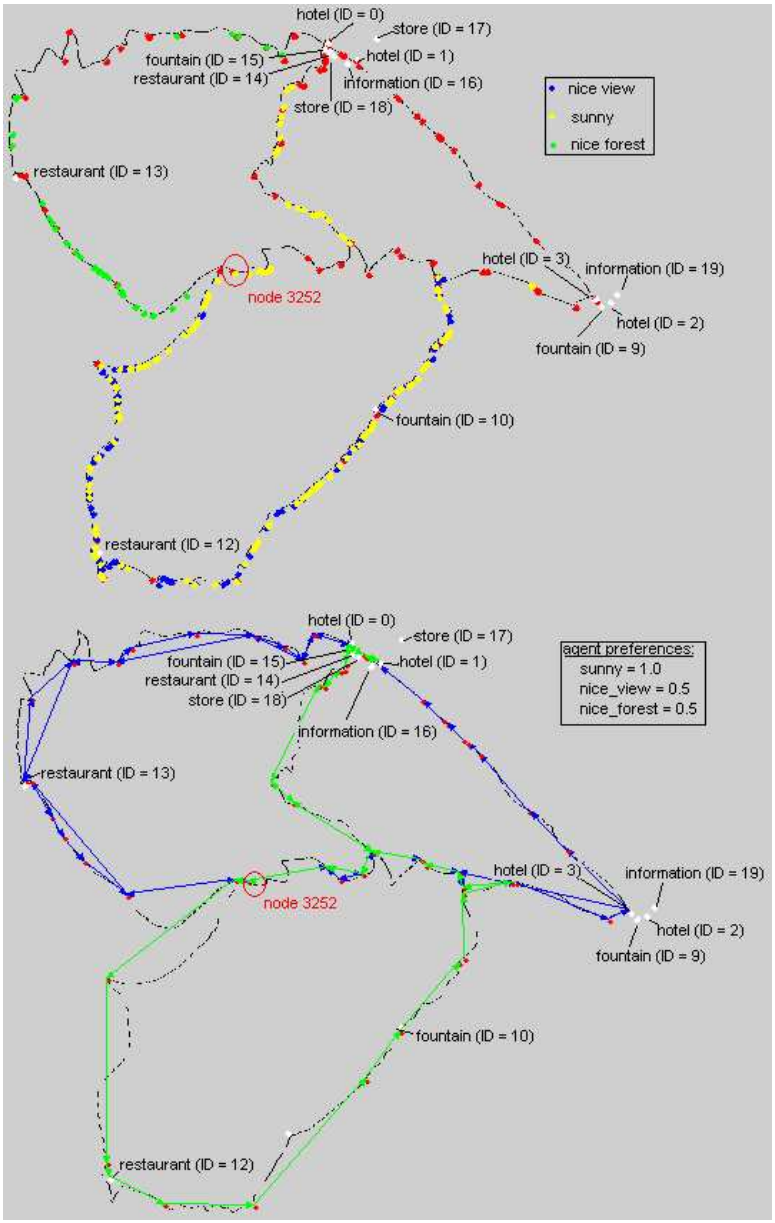


Abbildung 4.30: Oben in der Abbildung sind die Vorkommen der Ereignisse für die verschiedenen Attrakivitätstypen an den entsprechenden Koordinaten eingezeichnet. Der Agent nimmt drei verschiedene Arten von Attraktivitäten wahr: 'Sonnenschein' (gelbe Punkte), 'schöner Wald' (grüne Punkte) und 'schöne Aussicht (blaue Punkte). Die verschiedenen Attraktivitäten werden mit den entsprechenden Routenobjekten assoziiert, von denen aus sie wahrgenommen werden. Sie formen zusammen mit den Präferenzen des Agenten die zugehörigen Bewertungen. Aufgrund dieser und dem in Abbildung 4.29 gegebenen Aktivitätenvorschlag wird die Route im rechten Teil der Abbildung gefunden.

vitätsplan in Abbildung 4.29 ausführen. Sobald er den Wegpunkt mit der Identität 3352 erreicht hat, entsprechen die wahrgenommenen Attraktivitäten nicht mehr den erwarteten. Der Agent bemerkt dies, indem er die erwartete Bewertung der Routenobjekte im Plan mit der aktuellen Bewertung des entsprechenden Routenobjektes vergleicht. Die Abweichungen sind in Tabelle 4.3 gegeben.

Die maximale erlaubte Abbweichung von der Bewertung der restlichen Route ist hier

sunny	0.0
nice view	1.0
nice forest	1.0

Tabelle 4.3: Ab dem Wegpunkt 3352 bemerkt der Agent, dass der erwartete Wert für Sonnenschein nicht mit dem wahrgenommenen Wert übereinstimmt. Die Routenverzweigung ist in den Abbildungen 4.30 und 4.32 rot markiert. Die Tabelle zeigt die dort festgestellten Abweichungen der einzelnen Attraktivitätstypen. Ist die Abweichung 1.0 sind die Bewertungen gleich. Bei einer Abweichung von der Grösse 0.0 hat die Bewertung unendlich viel abgenommen. Fals die Abweichung 2.0 ist hat sie unendlich viel zugenommen.

durch einen Schwellenwert von 0.5 begrenzt. Dieser wird in diesem Fall überschritten:

$$\frac{|6.5764935740809305 - 0.0|}{0.0} > 0.5$$

Die entsprechende Übergangsmatrix und die daraus resultierenden bedingten Wahrschein-

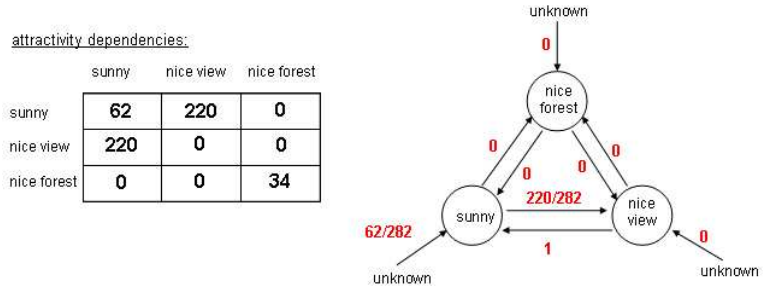


Abbildung 4.31: Links ist der Zustand der Übergangsmatrix gegeben, als der Wegpunkt 3352 erreicht wird, wo die Abweichung der entsprechenden Attraktivität vom Erwartungswert gemessen wurde. Der ij-te Eintrag wiederspiegelt wie oft $Attraktivität_i$ wahrgenommen wurde während $Attraktivität_j$ noch aktiv war oder umgekehrt. Rechts ist der Graph mit den entsprechenden Übergangswahrscheinlichkeiten gegeben, die nach Gleichung 2.11 berechnet werden.

lichkeiten, die in Untersektion 2.4.2 beschrieben werden sind in Abbildung 4.31 gegeben. Die Gewichte der Attraktivitäten werden dann nach Gleichung 3.11 folgendermassen berechnet:

$$weight_{sunny} = 0.0 \cdot (1.0 \cdot \frac{62}{220+62+0} + 1.0 \cdot \frac{220}{220+62+0} + 1.0 \cdot \frac{0}{220+62+0}) = 0.0$$

$$weight_{niceview} = 1.0 \cdot (0.0 \cdot \frac{220}{220+0+0} + 1.0 \cdot \frac{0}{220+0+0} + 1.0 \cdot \frac{0}{220+0+0}) = 0.0$$

$$weight_{niceforest} = 1.0 \cdot (0.0 \cdot \frac{0}{0+0+34} + 1.0 \cdot \frac{0}{0+0+34} + 1.0 \cdot \frac{34}{0+0+34}) = 1.0$$

Man stellt fest, dass sich nicht nur die Gewichtung von der Attraktivität 'Sonnenschein' verändert hat, sondern ebenso diejenige, von 'schöne Aussicht'. Dies war auch zu erwarten.

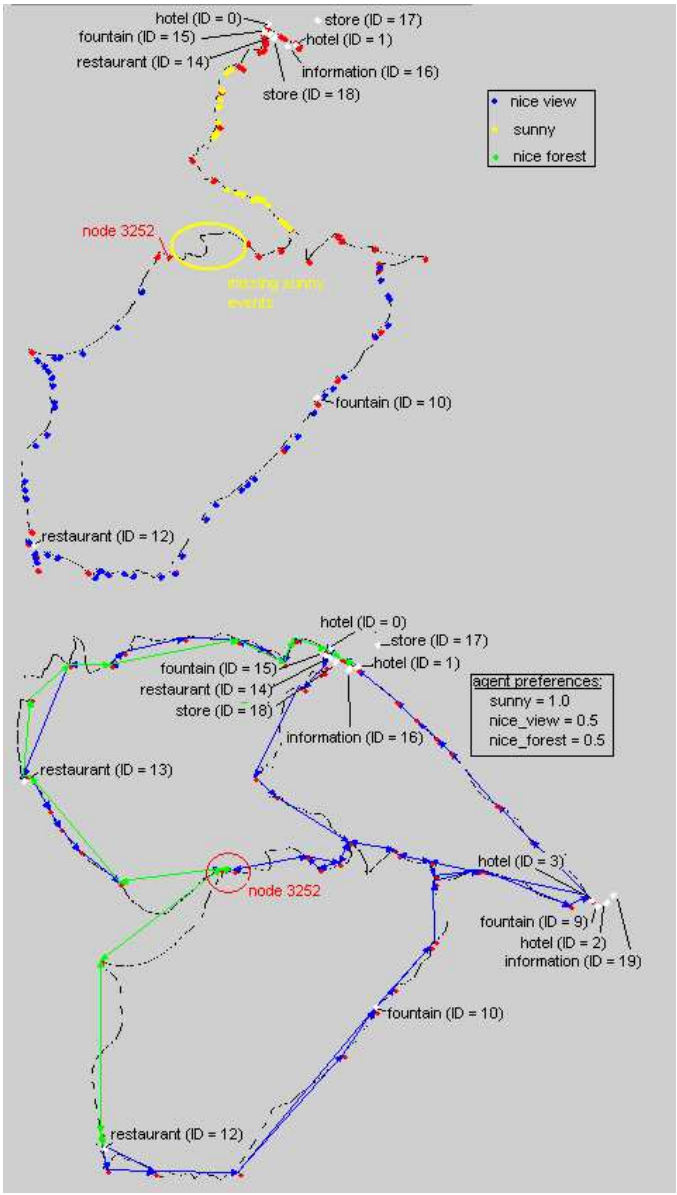


Abbildung 4.32: Oben in der Grafik sind die Ereignisse, die der Agent wahrnehmen würde, falls er die alte Route (Abbildung 4.30) ausführen würde, in einem Koordinatensystem abgebildet . Vergleicht man die Route mit den gegebenen Attraktivitätsvorkommen in Abbildung 4.30 oben, so stellt man fest, dass ab kurz vor dem Wegpunkt 3352 keine 'Sonnenschein' Ereignisse mehr auftreten. Der Agent bemerkt dies, sobald er den Wegpunkt 3352 erreicht hat. Er beginnt sofort eine neue Route zu suchen, die den geänderten Bedingungen Rechnung trägt. Der Agent findet die Route die im unteren Teil der Abbildung gegeben ist. Vergleicht man die Route mit dem vorkommen der verschiedenen Attraktivitätsereignisse in Abbildung 4.30 oben, zieht es der Agent nun vor, durch den Wald zu laufen.

Wenn wir nochmals die Abbildung 4.30 oben betrachten, kommt schöne Aussicht ausschliesslich zusammen mit 'Sonnenschein' vor. Verschwindet dieser, so glaubt der Agent an auch keine schöne Aussicht mehr wahrzunehmen. Dies ist eine Annahme aufgrund der bisherigen Erfahrungen. Nimmt der Agent später die schöne Aussicht nicht mehr nur in Abbhängigkeit von Sonne wahr, so wird die Übergangsmatrix, die die Abhängigkeiten der Attraktivitäten wiedergibt, entsprechend angepasst. Mit den neu berechneten Gewichten und den Bewertungen der Routenobjekte in der mentalen Karte wird eine neue Route für den restlichen Aktvitätenplan gesucht. Das Resultat ist in Abbildung 4.32 unten zu sehen. Der Agent bevorzugt nun eine Route, die mehr vom schönen Wald geprägt ist. Dieser Attraktivitätstyp ist nicht davon abhängig ist, ob die Sonne nun scheint oder nicht.

Anmerkung: Als Routingalgorithmus wird hier Dijkstra verwendet.

```
<request type="plan" agent="1">
    <plan>
        <act name="hotel" locationtype="object_ID" location="0" endtime="25028" />
        <act name="hike" duration="20498" />
        <act name="eat" duration="3600" score="5.0"/>
        <act name="hike" duration="1378" />
        <act name="buy_drink" duration="900" score="5.0"/>
        <act name="hike" duration="2855" />
        <act name="rest_and_sleep" start_time="54259" />
    </plan>
</request>
```

Abbildung 4.33: Im Experiment ist der Agent automatisiert. Jeden Tag wird er selbständig einen neuen Aktivitätenplan generieren. Die Abbildung zeigt ein Beispiel eines solchen Planes. Die Pläne die hier verwendet wurden können maximal 5 Aktivitäten plus eine Start- und Endaktivität beinhalten. Die Startaktivität entspricht jeweils der momentanen Position des Agenten.

4.2.7 Autonome Agenten

Die bisherigen Experimente beruhen auf vorgegebenen Datensätzen und sollen die bisher in der Theorie besprochenen Merkmale in der Simulation der mentalen Karte austesten. Sobald die Simulation startet, werden die Agenten gewisse Initialrouten ablaufen und sich dabei ein individuelles, räumliches Grundwissen aneignen. Danach generiert der Agent selbständig Aktivitätenpläne nach dem in Sektion 3.4 besprochenen Algorithmus. Er vervollständigt diese mit seinem mentalen Wissen und setzt sie in der Simulation um. Die Aktivitätenpläne sind vorerst für einen ganzen Tag ausgelegt. Wir gehen zusätzlich von unabhängigen, d.h. gegenseitig nicht beeinflussbaren Attraktivitäten aus. Es gibt kein 'Interdayreplanning'. Sobald der Agent den Plan erfüllt hat, generiert er auf seinem neuen, eventuell erweiterten Wissen, einen neuen Tagesplan für den nächsten Tag. Durch das fortlaufende Ausführen neuer Tagespläne bewegt sich der Agent autonom in seiner konstruierten Umgebung.

Da der Agent in einer abgeschlossenen Welt lebt wird erwartet, dass sein Wissen konvergiert. Dies muss nicht erst der Fall sein, wenn er die vollständige Umgebung erforscht hat. Konvergenz tritt ein, wenn der Agent für seine Aktivitäten zufriedenstellende Routen gefunden hat, bei deren Ausführung kein neues Wissen mehr gefunden wird.

Im folgenden Experiment soll gezeigt werden, dass das Wissen des Agenten konvergiert. Für die initiale mentale Karte wurden die Routeninformationen des Traingssets in Appendix A.1 verwendet. Der Agent führt automatisch generierte Plan aus. In Sektion 3.4 wurde gezeigt wie solche Aktivitätenpläne konstruiert werden. Der Agent kann maximal 5 Aktivitäten am Tag planen. Abbildung 4.33 gibt ein Beispiel eines solchen Plans. Die neu wahrgenommenen Informationen werden in die mentale Karte integriert. Danach beginnt der Agent von vorne. In jedem Durchgang wird die Anzahl Routenobjekte in der mentalen Karte gezählt. Die Zahl muss nach einer gewissen Anzahl Iterationen konvergieren. Bei

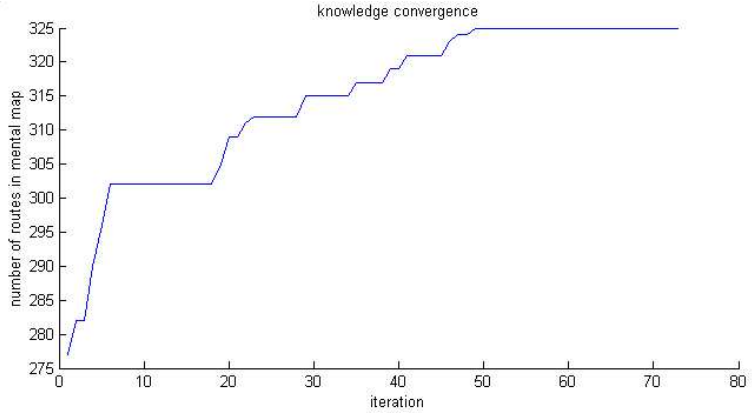


Abbildung 4.34: Für einen Agenten wurden 73 Tage simuliert. Der Agent hatte ein initiales mentales Wissen. Zufällig wurden am jede jedes Tages ein neuer Aktivitätenplan generiert und eine passende Route dazu gesucht. Mit den wahrgenommenen Informationen wurde die mentale Karte erweitert. In einer begrenzten Agentenwelt ist zu erwarten, dass der Agent die Gegend immer besser lernt. Mit anderen Worten wird das neu dazugewonnene Wissen wird nach jedem Tag kleiner. Hier wurde die Anzahl der Routenobjekte in der mentalen Karte gemessen. Sobald keine neuen Routenobjekte mehr wahrgenommen werden, wächst die mentale Karte nicht mehr. Der Agent verwendet nur noch die bereits vorhandenen Routenobjekte. Die Abbildung zeigt, dass das Wissen bei einem Stand von 325 Routen konvergiert.

Konvergenz hat der Agent die Umgebung genug erforscht, um alle Aktivitäten ausführen zu können. Mit anderen Worten, er lernt nichts mehr dazu. Das Ergebnis in Abbildung 4.34 zeigt, dass die Anzahl Routen in der mentalen Karte mit der Zeit konvergiert.

Konvergiert das Wissen, so muss es auch die Anzahl Routen in der mentalen Karte des

```
<request type="plan" agent="1">
    <plan>
        <act name="hotel" locationtype="object_ID" location="2" endtime="0" />
        <act name="hike" duration="8976" />
        <act name="eat" duration="3600" score="5.0"/>
        <act name="hike" duration="6023" />
        <act name="buy_drink" duration="900" score="5.0"/>
        <act name="hike" duration="6760" />
        <act name="get_local_information" duration="900" score="5.0"/>
        <act name="hike" duration="2514" />
        <act name="hotel" locationtype="object_ID" location="2" endtime="31243" />
    </plan>
</request>
```

Abbildung 4.35: Der in der Abbildung angegebene Aktivitätenplan wird mehrmals nacheinander auf die mentale Karte eines Agenten angewandt. Mit jeder Iteration wird erwartet, dass die resultierende Route aufgrund des dazugewonnen Wissens etwas ändert. Sobald kein neues Wissen mehr gelernt wird konvergiert die Route.

Agenten tun. Mit anderen Worten, falls sich das Wissen nicht mehr ändert, wird der Agent für einen Aktivitätenplan immer dieselbe Route generieren. In einem Experiment wird basierend auf derselben mentalen Karte wie im vorherigen Experiment jeweils der gleiche Plan ausgeführt. Es wird erwartet, dass sich die Route aufgrund des dazugewonnen Wissen ändert. Soblad die Ausführung der gefundenen Route kein neues Wissen mehr bringt, muss sie konvergieren. In Abbildung 4.35 ist der entsprechende Aktivitätenplan gegeben. In Abbildung 4.36 die Bewertung der Route, die jeweils für einen bestimmten Tag gefunden wurde. Nach ungefähr 30 Tagen hat die Route konvergiert. Die Punktzahl bleibt von da an konstant.

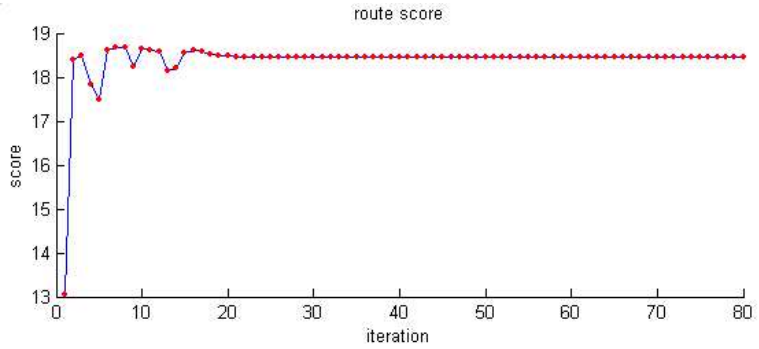


Abbildung 4.36: In einem Experiment wird auf einem bestimmten mentalen Grundwissen immer wieder deselbe Aktivitätenplan ausgeführt. Die Abbildung zeigt die Bewertung der besten gefundenen Route. Diese konvergiert nach gut 30 Tagen. Ab diesem Zeitpunkt verändert sich die Route nicht mehr. Der Grund ist, dass der Agent nun kein neues Wissen mehr dazugewinnt. Die anfänglichen Oszillationen in der Routenbewertung sind darauf zurückzuführen, dass der Routingalgorithmus auf dem mentalen Wissen approximativ ist (vgl. Sektion 4.2.1

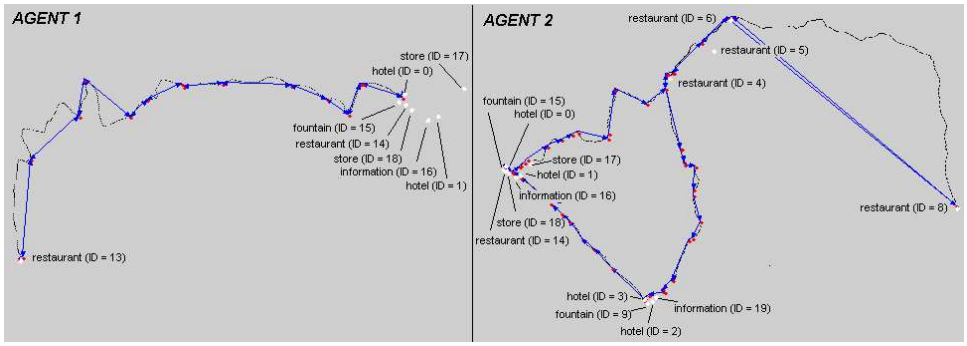


Abbildung 4.37: Links ist die mentale Karte von Agent 1 gegeben und rechts diejenige von Agent 2. Die beiden mentalen Karten überschneiden sich nur minimal. Die Überschneidung ist lokal um das Objekt von Typ 'Hotel' mit der Identität 0 angelegt.

4.3 Kommunikation zwischen Agenten

Bisher wurde von einem einzigen Agenten ausgegangen. Die Simulation wurde so konstruiert, dass diese einfach auf mehrere Agenten erweitert werden kann. Jeder Agent hat sein eigenes mentales Wissen, dass er selbständig erlernt hat. Durch Kommunikation dieser Daten untereinander eröffnen sich neue Möglichkeiten das eigene Wissen zu vergrössern (vgl. Sektion 3.5).
Die Agenten in der Simulation kommunizieren folgendermassen: Sie schicken allen Agenten, die sich am Ende eines Tages auf dem selben Objekt befinden, einen Aktivitätenplan. Die Agenten probieren, diesen auf das eigene mentale Wissen anzuwenden. Der anfragende Agent macht dasselbe. Diejenigen Agenten, die eine Route für den Atkivtätenplan gefunden haben, schicken diese dem Agenten zurück. Zusätzlich erhält der Agent ebenfalls die Bewertung des anderen Agenten. Die Bewertung jedes Agenten ist dabei individuell. Sie hängt von den persönlichen Präferenzen ab. Der Agent wählt die beste Route und führt sie aus. Ein Agent kommuniziert zufällig oder falls er keine Lösung für ein Problem findet. Dazu muss sich ein anderer Agent gerade am selben Ort befinden.
Die Kommunikationsfähigkeit soll in einem Experiment getestet werden. Dabei werden zwei Agenten verwendet, die vom selben Objekt aus starten. Die mentalen Karten der

```
<request type="plan" agent="1">
    <plan agent="1" score="0.4023804779391629 3" >

        <act name="hotel" location_type="object_id" location="0" end_time="0" />

        <act name="hike_start" location_type="object_id" location="0" expected_time="3638" start_time="0" />
        <act name="hike_waypoint" location_type="node_id" location="4086" />
        <act name="hike_waypoint" location_type="node_id" location="4125" />
        <act name="hike_waypoint" location_type="node_id" location="4122" />

          ...

        <act name="hike_waypoint" location_type="node_id" location="3886" />
        <act name="hike_waypoint" location_type="node_id" location="3593" />
        <act name="hike_end" location_type="object_id" location="13" end_time="3638" />

        <act name="eat" location_type="object_id" location="13" start_time="3638" end_time="5438" />

        <act name="hike_start" location_type="object_id" location="13" expected_time="3637" start_time="5438" />
        <act name="hike_waypoint" location_type="node_id" location="3593" />
        <act name="hike_waypoint" location_type="node_id" location="3886" />

          ...

        <act name="hike_waypoint" location_type="node_id" location="4125" />
        <act name="hike_waypoint" location_type="node_id" location="4086" />
        <act name="hike_end" location_type="object_id" location="0" end_time="9075" />

        <act name="hotel" location_type="object_id" location="0" start_time="9075" />

    </plan>
</request>
```

Abbildung 4.38: Bewegt sich bloss ein einzelner Agent in der Agentenwelt, ist es ihm nicht möglich zu kommunizieren. Nach dem die mentale Karte wie Abbildung 4.37 dargestellt fertig aufgebaut ist, führt der Agent den Aktivitätenplan aus Abbildung 4.14 aus. Die resultierende Route ist in der Form eines Planes in der Abbildung gegeben. Die dabei erwartete Bewertung ist ungefähr 0.40.

```
<request type="plan" agent="1">
    <plan agent="1" score="0.9072690704217732" >

        <act name="hotel" location_type="object_id" location="0" end_time="0" />

        <act name="hike_start" location_type="object_id" location="0" expected_time="13893" start_time="0" />
        <act name="hike_waypoint" location_type="node_id" location="4086" />
        <act name="hike_waypoint" location_type="node_id" location="4058" />

        ...

        <act name="hike_waypoint" location_type="node_id" location="5316" />
        <act name="hike_waypoint" location_type="node_id" location="3792" />
        <act name="hike_end" location_type="object_id" location="5" end_time="13893" />

        <act name="eat" location_type="object_id" location="5" start_time="13893" end_time="15693" />

        <act name="hike_start" location_type="object_id" location="5" expected_time="4492" start_time="15693" />
        <act name="hike_waypoint" location_type="node_id" location="5304" />
        <act name="hike_waypoint" location_type="node_id" location="5047" />

        ...

        <act name="hike_waypoint" location_type="node_id" location="4058" />
        <act name="hike_waypoint" location_type="node_id" location="4086" />
        <act name="hike_end" location_type="object_id" location="0" end_time="20185" />

        <act name="hotel" location_type="object_id" location="0" start_time="20185" />

    </plan>
</request>
```

Abbildung 4.39: Der Agent 1 schickt Agent 2 seinen Aktivitätenplan. Die Route zum Aktivitätenplan, die Agent 1 ausgibt, geht via das Objekt vom Typ 'Restaurant' mit der ID = 5. Die mentale Karte von Agent eins enthält kein solches Objekt. Folglich muss die Information vom zweiten Agenten stammen. Die mentale Karte von Agent 2 enthält das entsprechende Objekt.

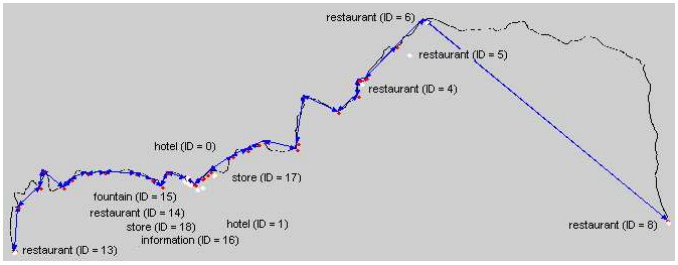


Abbildung 4.40: Führt der Agent den Plan des anderen Agenten aus vergrössert sich dabei sein eigenes mentales Wissen. Die neuen Informationen können in der mentalen Karte des anderen Agenten wiedergefunden werden.

Agenten sind in Abbildung 4.37 gegeben. Sie unterschieden sich stark und überlappen nur in der Region um das Objekt von Typ 'Hotel' mit der Identität 0. Das gemeinsame Startobjekt ist vom Typ 'Hotel' und hat die Identität 0. Damit die Agent Kommuniziert wird die definierte Variable für die Kommunikationswahrscheinlichkeit auf 1.0 gesetzt (vgl. Tabelle A.2).
In einem ersten Durchgang soll nur Agent eins verwendet werden. Nachdem er die mentale Karte aufgebaut hat soll er den Aktivitätenplan der Abbildung 4.14 auf sein mentales Wissen anwenden. In einem zweiten Durchgang werden Agent eins und Agent zwei gleichzeitig in der Simulation eingesetzt. Die Agenten starten beide im Objekt vom Typ 'Hotel' und der Identität 0. Agent eins versucht wiederum denselben Aktivitätenplan mit räumlicher und zeitlicher Information zu komplettieren. Es wird erwartet, dass der Agent merkt, dass ein weiterer Agent anwesend ist und diesen Anfragt bevor er den eigenen Plan ausführt. Falls der Plan, der auf dem mentalen Wissen des anderen Agenten basiert besser ist, soll er diesen verwenden.
In Abbildung 4.38 ist der resultierende Plan aus dem ersten Durchgang des Experiments gegeben. Die erwartete Bewertung der Route ist ungefähr 0.40. Im zweiten Teil des Experiments, findet Agent eins jedoch eine Route, die eine erwartete Bewertung von ungefähr 0.90 hat. Der XML-Plan ist in Abbildung 4.39 gegeben. Dieser muss aus der Anfrage an den zweiten Agenten stammen, zumal sie via das Objekt vom Typ 'Restaurant' mit der Identität 5 geht, dass nur im mentalen Wissen von Agent zwei vohanden ist. Das Resultat ist also so wie erwartet.
Führt der Agent 1 den Plan plan von Agent 2 aus, so erweitert er mit den neu wahrgenommenen Daten die eigene mentale Karte. In Abbildung 4.40 ist das erweiterte Wissen graphisch gegeben. Vergleicht man diese neue mentale Karte mit derjenigen vor der Ausführung des Plans in Abbildung 4.37 links, so stellt man fest, dass die Karte erweitert wurde mit den wahrgenommenen Daten der ausgeführten Route via das Objekt von Typ Restaurant' und der ID=5.
Durch das Verwenden von Kommunikation, kann der Agent das eigene Wissen schneller vergrössern. Dies wurde anhand eines Experiments mit drei Agenten getestet. Zum Aufbauen des mentalen Initialwissens dienten die Daten von je 3 unterschiedlichen Routen aus den Trainingsrouten in Appendix A.1. Die Ausdehnung des mentalen Wissens wird anhand der Anzahl der Routenobjekte, die in einem Durchgang neu zum mentalen Wissen dazukommen gemessen. Das Resultat ist in Abbildung 4.41 gegeben. Es wurde eine Kommunikativität von 0.6 verwendet. Das selbe Experiment haben wir bereits mit einem einzelnen Agenten in Untersektion 4.2.7 durchgeführt. Die Abbildung 4.34 zeigt das Resultat. Vergleicht man die beiden Kurven, kann man feststellen, dass die Anzahl Routen hier bei den drei untereinander kommunizierenden Agenten, von Iteration zu Iteration, in Durschnitt schneller zunimmt. Eine Iteration entspricht dabei einem simulierten Tag. Wie im Experiment mit einem Agenten, muss die Anzahl Routen bei jedem Agenten konvergieren. Aufgrund technischer Probleme von ALPSIM bei der gleichzeitigen Ausführung

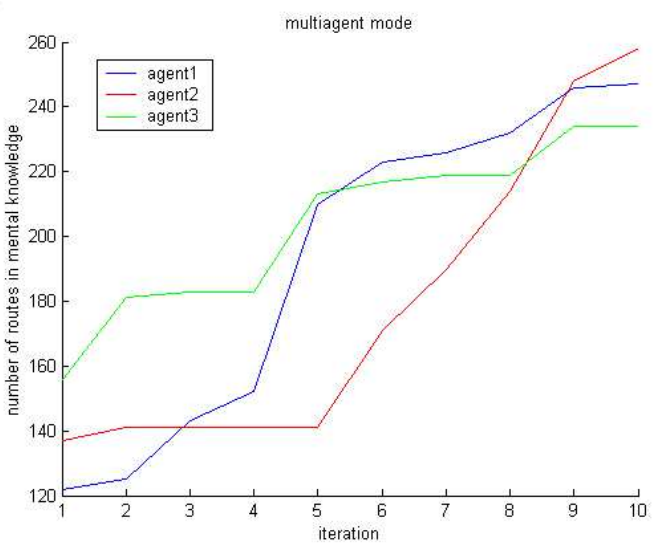


Abbildung 4.41: Die Abbildung zeigt, die Entwicklung der Anzahl Routen im mentalen Wissen für 3 Agenten. Diese Zahl soll die Grösse des Wissens wiedergeben. Jeden Tag wollen die Agenten bestimmte Aktivitäten ausführen. Um eine entsprechende Route für die Aktivitäten, die sie ausführen möchten zu finden, können die Agenten hier auch miteinander kommunizieren. Sie schicken dazu einem anderen Agenten den Aktivitätenplan. Dieser vervollständigt ihn mit Hilfe des eigenen mentalen Wissens. Mit nur einem einzelnen Agenten in der Simulationswelt ist es nicht möglich zu kommunizieren. Die einzige Möglichkeit das mentale Wissen dort zu vergrössern sind 'visuelle Links'. In Abbildung 4.34 ist das entsprechende Experiment gegeben. Beim Vergleichen erkennt man, dass der Zugewinn an Routeobjekten bei einem einzelnen Agenten von Iteration zu Iteration hier im Durchschnitt kleiner ist.

mehrerer Routenpläne, konnten nicht genügend Tage simuliert werden, um dies zu zeigen.

Kapitel 5

Diskussion der Resultate und Ausblick

5.1 Zusammenfassung

Die Agenten in der Simulation verwenden individuell wahrgenommene visuelle Informationen, um eine mentale Karte aufzubauen. Die mentale Karte umfasst die direkt erfahrene räumliche Umgebung des Agenten. Die Daten werden dazu verwendet, eine Route zu einem Aktivitätenplan zu finden. Der Aktivitätenplan selbst ist eine Abfolge von punktuellen und räumlich ausgedehnten Aktivitäten. Welche Objekte zu welchen Aktivitäten passen, ist durch eine Generalisationshierarchie gegeben. Dieses Wissen ist im Agenten fest eingebaut.
Das mentale Wissen ist so arrangiert, dass schnell auf andere räumlich lokale Informationen zugegriffen werden kann. Es existiert auf drei verschiedenen Levels. Der Agent nimmt Objekte wahr und bewertet diese. Für jedes Objekt entsteht ein Bewertungssignal über die Zeit. Die Bewertung nimmt bei jedem Vorkommen etwas zu. Eine Zerfallsfunktion verringert die Bewertung über die Zeit. Objekte, die gleichzeitig wahrgenommen werden und deren Bewertung ein Schwellenwert übersteigt, werden zu Ansichten zusammengefasst. Diese wird permanent gespeichert. Der Schwellenwert passt sich den zuletzt wahrgenommenen Objekten an. Aufeinanderfolgende Ansichten sind durch einen 'Action Link' verbunden. Eine Abfolge von 'Action Links', die zwei aufeinanderfolgend wahrgenommene Verzweigungen des Strassennetzwerks verbinden, definieren ein Routenobjekt.
Der Agent nimmt an bestimmten Orten im Raum gewisse Attraktivitätsereignisse wahr. Eine Attraktivität beeinflusst den emotionalen Zustand des Agenten. Die persönliche Präferenz für einen Attraktivitätstyp und deren Häufigkeit während ein Routenobjekt abgelaufen wird, bestimmt dessen Bewertung. Die Bewertung eines Routenobjekts setzt sich zusammen aus verschiedenen Attraktivitätstypen.
Das mentale Wissen kann als Graph betrachtet werden dessen Kanten Routenobjekte und Knoten Routenverzweigungen sind. Um einen Aktivitätenplan zu erfüllen, muss der Agent eine Route finden, um die verschiedenen Objekte zu erreichen, an denen er eine punktuelle Aktivität ausführen möchte. Die Fortbewegung auf einem Weg ist definiert als Aktivität mit räumlicher Ausdehnung. Das Ziel ist es, ein Weg zu finden, der optimale Bewertung hat und die der Aktivität zur Verfügung stehende Zeit möglichst genau einhält. Der verwendete Algorithmus ist A* oder Dijkstra. Um den Aktivitätenplan zu erfüllen, werden alle Kandidatenobjekte für eine punktuelle Aktivität rekursiv durchprobiert. Voraussetzung ist, dass eine gültige Route existiert zum Objekt. Die Aktivitätenpläne werden zufällig generiert. Dabei soll eine Vorliebe einer Aktivität für eine bestimmte Tagesaktivität erhalten bleiben.
Beim Ausführen eines Aktivitätenplans gewinnt der Agent neue Informationen. Dies ist insbesonders möglich, indem Daten, die er nur visuell wahrgenommen hat, in die Routen-

planung mit einbezieht. Die entsprechende Route ist intern als ein visueller Link gegeben. Ein visueller Link ist in einer Ansicht festgehalten. Der Ursprung ist also in einem bereits begangen Ort. Bei der Ausführung wird der Link mit einer Route vervollständigt. Für den Agenten kommt dies mit dem Nachschauen auf einer Karte gleich.
Die bereits bekannten Informationen werden wiedererkannt. Wird eine neue Ansicht wahrgenommen, vergleicht der Agent diese mit allen ihm bereits bekannten Ansichten mit gleichem Ursprungsobjekt und gleicher Perspektive. Ein Ursprungsobjekt ist meistens eine Strasse, eine Verzweigung im Strassennetzwerk oder ein Objekt, das begangen wird. In Ansichten mit gleicher Perspektive haben die gleichen Objekte jeweils die gleiche Anordnung. Zusätzlich wir deren Ähnlichkeit geprüft. Hat das ähnlichste Vergleichsobjekt, einen Ähnlichkeitswert über einem bestimmten Schwellenwert, wird es als gleich eingestuft. Der Vergleich beruht auf einem Alignment der Topologie, d.h. Reihenfolge der wahrgenommenen Objekte. Der Algorithmus der verwendet wurde ist 'Dynamisches Programmieren'.
Die Simulation kann auf mehrere Agenten ausgedehnt werden. Dadurch eröffnet sich eine neue Möglichkeit das mentale Wissen auszuweiten. Die Agenten können Informationen austauschen indem sie einen Aktivitätenplan von einem anderen Agenten vervollständigen lassen. Ist die Route die der andere Agent findet, besser als die eigene, so wird diese ausgeführt werden. Die Bewertung der Route ist jedoch mit Vorsicht zu geniessen, da der andere Agent andere Präferenzen haben kann. Dieselbe Route hätte für die beiden Agenten dann eine unterschiedliche Bewertung.
Die mentale Karte ist in der Simulation automatisiert. Jeder Agent hat eine initiale mentale Karte. Der Agent beginnt mit einem Tagesplan, führt ihn aus, generiert einen neuen, etc. .Durch vergrössern des Wissen sollten sich auch die Routen verbessern. Die Verbesserung hängt davon ab, wie viel der Agent kommuniziert und visuelles Wissen in der Planung der Route verwendet.
Die mentale Karte kann mit dynamisch veränderbaren Bewertungen umgehen. Der Agent merkt sich wie stark die einzelnen Aktivitätstypen von einander abhängen. Ist die Bewertung bei der Ausführung der Route eines Aktivitätenplans ungleich der erwarteten Bewertung in der mentalen Karte, initiert der Agent ein 'Replanning'. Für die restlichen Aktivitäten wird aufgrund der neuen Erwartungen eine andere Route gesucht. Die neuen Erwartungen sind dabei beeinflusst durch die verändert wahrgenommenen Attraktivitätstypen. Diese beeinflussen auch die Bewertungen der davon abhängigen Attraktivitätstypen.
Die letzte Funktionalität konnte nur an einem konstruierten Beispiel getestet werden. Der Grund ist, dass der Simulationsteil, der die Wahrnehmungsevents des Agenten generiert, keine solchen dynamischen Veränderungen automatisch generiert.

5.2 Diskussion der Resultate

In Kapitel 4 sind verschiedene Testresultate der mentalen Karte gegeben. Die Testdaten basieren auf GIS-Daten der Region Gestaad. Den meisten Experimenten liegen die Testrouten aus Appendix A.1 zu Grunde. Die Basiseinstellungen der Simulation ist in Appendix A.2 gegeben.
In Sektion 4.1 wird der Aufbau der Datenstruktur getestet. Zuerst wird die Signalgeneration für einzelne wahrgenommene Objekte geprüft. Danach wird gezeigt wie mit Hilfe von Parametern die Datenreduktion beeinflusst werden kann. Das Ergebnis hat direkten Einfluss auf die Anzahl wahrgenommenen Ansichten. Es wird illustriert, dass die Wahrnehmung auf einer Route asynchron ist. Auf dem Hinweg einer Route wird nicht gleich wahrgenommen wie auf dem Rückweg.
Das Strassennetzwerk der Simulation hat eine zu geringe Auflösung. Objekte, wie Häuser sind nicht direkt daran angebunden. In der Simulation werden interne Routen generiert. Wie die Objekte genau mit dem Strassennetzwerk verbunden werden ist nicht Aufgabe der mentalen Karte, sondern der ausführenden Simulation. Es wird gezeigt, dass entsprechende interne Links erzeugt werden. Eine Route ist anhand der verschiedenen Verzweigungsob-

jekten charakterisiert. Damit der Agent solche Routenobjekte erkennt, muss er eine Ansicht auf den entsprechenden Verzweigungen erkennen. Die Generierung von visuellen Daten wird von der Simulation an einem solchen Ort erzwungen. Das Erkennen der Ansicht setzt voraus, dass einerseits visuell wahrnehmbare Objekte existieren und anderseits deren Bewertung genug gross ist, um in die momentane Ansicht aufgenommen zu werden. Anhand von zwei Beispielen wird gezeigt, wo solche Ansichten erkannt werden und wo nicht. Eine Ansicht, die nicht erkannt wird, ist nicht weiter schlimm. Der Agent hat dann die Verzweigung für die Aktivitätenplanung einfach nicht zur Verfügung. In einem weiteren Experiment wird illustriert wie Attraktivitätenvorkommen in Bewertungen von Routenobjekten umgesetzt werden. Schliesslich wird die Wiedererkennungsrate von Ansichten im Speicher untersucht, falls eine Route mehrmals abgelaufen wird. Die neuen Ansichten nehmen exponentiell ab.
Die Tests zeigen, dass der Aufbau der Datenstruktur funktioniert. Die Voraussetzung, dass irgendein Objekt erkannt wird, sind visuelle Objekte. Ohne sie ist der Agent sozusagen blind. Es müssen genügend solche Objekte in der Simulationswelt vorhanden sein. Je weniger Objekte vorhanden sind, desto schlechter kann sich der Agent die Umgebung merken.
In Sektion 4.2 werden zuerst zwei Routingalgorithmen gegeneinander abgewogen. Algorithmus eins beruht auf Dijkstra, Algorithmus zwei auf A*. Die Aufgabe ist es, eine Route zwischen zwei Objekten zu finden. Es ist eine Zeitlimite vorgegeben und Routenobjekte mit unterschiedlicher Bewertung. Die Route soll die Bewertung der verwendeten Routenobjekte maximieren und darf eine vorgegebene Abweichung von der Zeitlimite nicht überschreiten. Der optimale Trade-Off von Zeitabweichung und Bewertung maximieren ist gefragt. Die Bewertung einer Route wird mit dieser Abweichung gewichtet. Im Experiment ist A* erfolgreicher, da er zusätzliche das Zeitlimit in die Routenbewertung mit einbezieht. Sonst sind die Algorithmen gleich. Dijkstra ist mehr 'greedy' kann jedoch zufällig besser sein als A*. Es muss bemerkt werden, dass beide Algorithmen, wegen den zwei Bedingungen die gleichzeitig optimiert werden müssen, nur Approximationen an die optimale Lösung liefern. Beide Algorithmen liefern in relativ kurzer Zeit eine 'gute' Lösung. Tendenziell verbessern sich die Bewertungen der Route für einen Aktivitätenplan, falls das mentale Wissen erweitert wird. Es kann jedoch auch vorkommen, dass es sich verschlechtert. Ein Grund ist, dass in beiden Algorithmen Knoten blockiert werden können von Routen die das maximale Zeitlimit überschreiten. Eine mindere Route wird dann bevorzugt. A* versucht das Problem zu verbessern. In A* ist das Problem dann auf die Güte der initialen Zeitschätzung an die optimale Route zurückzuführen.
In einem weiteren Experiment wird gezeigt, wie sich die Präferenzen eines Agenten für bestimmte Attraktivitätstypen auf die Routenwahl auswirken. Die Bewertung eines Routenobjekts basiert auf den örtlichen Vorkommen der Attraktivitäten und der Präferenz. Eine Route zu einem Aktivitätenplan ist den Präferenzen des Agenten angepasst.
Die Erweiterung des mentalen Wissens basiert einerseits auf der Verwendung von visuellen Daten, anderseits auf der Kommunikation zwischen den Agenten. In zwei Experimenten wird gezeigt, wie mit Hilfe der entsprechenden Techniken das mentale Wissen erfolgreich erweitert werden kann.
Falls der Agent eine Aktivität nicht erfüllen kann, wird diese übersprungen. Ein Beispiel veranschaulicht dies. Zusätzlich ist der Planungsalgorithmus so ausgelegt, dass auch mehrere Aktivitäten am Gleichen Ort ausgeführt werden können. Falls dies nicht möglich ist wird eine Aktivität eingefügt, die es erlaubt ein anderes Objekt zu suchen für die zweite Aktivität. Es ist anzumerken, dass letzteres nicht in die bisherige Aktivitätenplanung integriert ist. Ein Aktivitätenplan ist eine Abfolge von punktuellen Aktivitäten und solcher mit räumlicher Ausdehnung.
In einem Beispiel wird die Funktionalität von 'Interday Replanning' ausgetestet. Es muss hier angemerkt werden, dass die Simulation keine dynamischen Veränderungen der Vorkommen von Attraktivitäten erzeugen kann. Das Experiment basiert auf konstruierten Daten. Es soll zeigen, dass die mentale Karte volle Funktionalität in diesem Bereich hat. 'Inter-Day-Replanning' funktioniert also nicht vollständig, da die Simulation immer die gleiche

Verteilung der Attraktivitäten in der Simulationswelt benutzt.
Jeder Agent ist automatisiert, indem er einen Aktivitätenplan erzeugt, die dazu passende Route findet, sie ausführt und wieder von vorne beginnt. Es können mehrere Agenten gleichzeitig aktiv sein. Das 'Replanning' wird initiiert, sobald der Tag zu Ende ist. Experimente zeigen, dass das Wissen des Agenten dann konvergiert, sobald er bei der Ausführung von Routen kein neues Wissen mehr dazulernt. Für einen einzelnen Agenten wurden bis zu 140 Tagen nacheinander simuliert. Im Multiagentmodus treten technische Probleme auf bei der bisherigen Simulation ALPSIM, nachdem bis zu zehn Tage erfolgreich simuliert wurden. Die Simulation läuft zwar, liefert jedoch keine Wahrnehmungsevents nach einer bestimmten Anzahl Durchgänge. Die geschickten Pläne sind korrekt, da dieselben manuell erfolgreich ausgeführt werden können. Test waren daher nur sehr beschränkt möglich.
Ein grosses Problem beim Testen der Simulation war, dass nur wenige Testdaten zur Verfügung standen. Dies hat den Grund, dass die Generation von Wahrnehmungsevents in der entsprechenden Simulationskomponente lange nicht fehlerfrei funktionierte. Die Entwicklung dieser Komponente ist nicht in diesem Projekt enthalten.
Zusammenfassend kann gesagt werden: Es steht eine funktionsfähige Simulation zur Verfügung. Jeder Agent hat ein räumliches Wissen, dass auf persönlichen Erfahrungen beruht. Ein Agent kann einen Aktivitätenplan erzeugen und ihn auf seinem Wissen ausführen. Dabei werden zeitliche Bedingungen und die persönlichen Vorlieben des Agenten berücksichtigt. Die Agenten erweitern das Wissen autonom.

5.3 Ausblick

Die Simulation bisher liefert ein funktionierendes System, dass aufgrund von verschiedenen Wahrnehmungsevents ein internes Abbild, eine mentale räumliche Karte, der Simulationsumgebung lernt. Die Wiedererkennung von Informationen basiert dabei auf visuellen Ansichten. Der Agent verwendet die räumlichen Informationen intern um einen Aktivitätenplan auf die Umgebung abzubilden. Das Resultat ist eine Route auf der die Aktivtäten ausgeführt werden können.
Die Planung basiert bisher auf Routenknoten und Objekten, die allen Simulationskomponenten soweit wie nötig bekannt sind. In Zukunft kann man sich vorstellen, dass der Agent internen Knoten generiert. Solche Knoten können z.B. Orte sein wo die Aussicht besonders schön ist. Diese würden dann in die Routenplanung integriert.
Die Aktivitätenplanung ist bisher zufällig und von vorgegebene Informationen abhänging. Es ist offensichtlich, dass ein Individuum dazu Informationen aus der lokalen Umgebung mit einfliessen lässt. Der Agent könnte z.B. sobald er eine neue Bar entdeckt, Lust bekommen dort ein Bier trinken zu gehen.
Die Planungsalgorithmen, die bisher verwendet wurden, sind nicht optimal. Es ist schwer, eine optimale Route zu finden, die sich möglichst genau an ein vorgegebenes Zeitlimit hält und gleichzeitig eine Route findet, die möglichst den Präferenzen des einzelnen Agenten entspricht. Die Ausgangslage würde sich zusätzlich verkomplizieren, wenn weitere realistische Abhängigkeiten miteingebaut würden. Momentan macht es keinen Unterschied, ob ein Restaurant nachts um drei besucht wird oder Mittags um 12. Die Simulation beinhaltet zwar zeitabhängige Attraktivitäten, der Planungsalgorithmus vernachlässigt der Einfachheit halber diese Zeitabhängigkeit wieder. Eine solche von der Tageszeit abhängige Attraktivität kann dazu führen, dass nicht alle Objekte an einer bestimmten Tageszeit gleich attraktiv sind. Anderseits kann man sich vorstellen, dass eine Aktivität grundsätzlich an bestimmten Tageszeiten eine hohe Attraktivität hat. Ein Beispiel wäre essen, das bevorzugt am Morgen, Mittag und Abend ausgeführt wird.
Die automatische Hierarchisierung von Informationen würde eine Planung auf mehreren abstrakten Ebenen ermöglichen. Im Moment geschieht dies via das Generalisationswissen. Ein Individuum baut sich dieses Wissen selbständig auf. Ein solches strukturiertes Wissen würde die Komplexität der Vernetzung der Informationen erhöhen. Jedoch würde das

Wissen auch erlauben, Daten besser wiederzufinden. Bisher wird die Informationshierarchisierung nur sehr limitiert eingesetzt.
Die mentale Karte besitzt eine Form von 'Inter-Day-replanning'. Da die Simulationswelt eine statische Umgebung ist, die an einer Position immer die gleichen Wahrnehmungsevents sendet, wird nie ein solches 'Replanning' ausgelöst. Die bisherigen Datensätze zum Austesten sind konstruiert. Durch Miteinbezug von solchen dynamischen Veränderungen könnte automatisiertes 'Inter-Day-Replanning' ausgetestet werden. Auf der Seite der mentalen Karte sollte dafür die Struktur vorhanden sein und funktionieren.
Die Simulation kann auf mehrere Agenten ausgedehnt werden. Da für jeden Agenten eine enorme Menge an visuellen Informationen verarbeitet werden muss, verlangsamt sich das Programm sehr stark. Die Simulation müsste für eine bessere Performanz optimiert werden oder verteilt werden. Die 'Bottlenecks' sind vor allem die Menge an kommunizierten Daten pro Agenten.
Weiter wäre es interessant die Simulation mit grösseren Datensets auszutesten. Dies beinhaltet im Wesentlichen mehr visuelle Objekte die wahrgenommen würden. Dadurch wird die Umgebung realistischer.

Anhang A

Initialisierung der Simulation

A.1 Trainingspläne

Anfangs wird der Agent mit den XML-Daten einer bestimmten Anzahl ausgeführten Plänen, die die Simulation bei der Ausführung generiert initialisiert. Dieses initiale Wissen wird vom Agenten selbständig erweitert. Das individuell dazugewonnene Wissen charakterisiert die Wissensbasis eines Agenten. Die verwendeten Pläne starten und enden jeweils im selben Objekt.

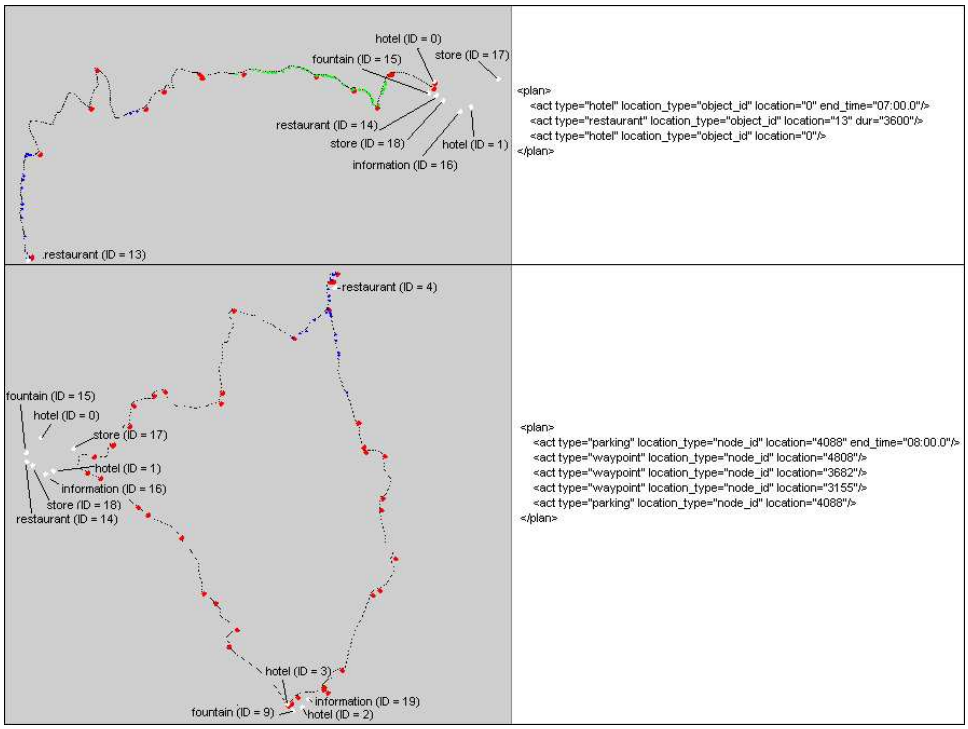


Abbildung A.1: Routen 1 und 2. Links sind die Daten, welche der Agent wahrnimmt in kartographischer Form zu sehen. Die weissen beschrifteten Punkte sind Objekte die der Agent sehen kann. Die Roten Punkte sind Verzweigungen des Strassennetzwerks. Die grünen und blauen Punkte, stehen für Orte wo der Agent schöne Aussicht oder schöner Wald wahrnimmt. Die schwarzen Punkte verdeutlichen die eigentliche Route. Die Pläne sind rechts in XML Form gegeben.

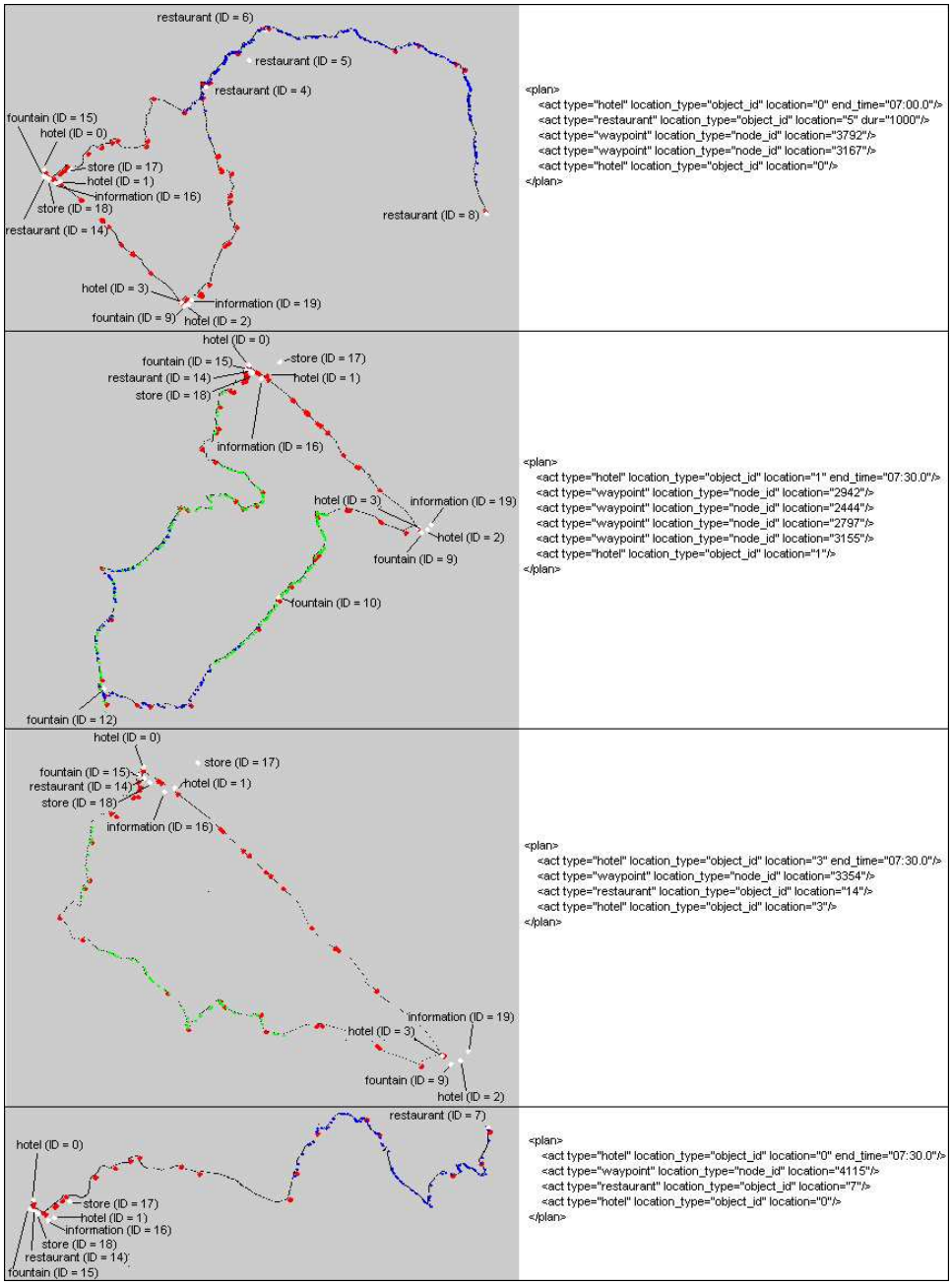


Abbildung A.2: Routen 3 bis 6. Links sind die Daten, welche der Agent wahrnimmt in kartographischer Form zu sehen. Die weissen beschrifteten Punkte sind Objekte die der Agent sehen kann. Die Roten Punkte sind Verzweigungen des Strassennetzwerks. Die grünen und blauen Punkte, stehen für Orte wo der Agent schöne Aussicht oder schöner Wald wahrnimmt. Die schwarzen Punkte verdeutlichen die eigentliche Route. Die Pläne sind rechts in XML Form gegeben.

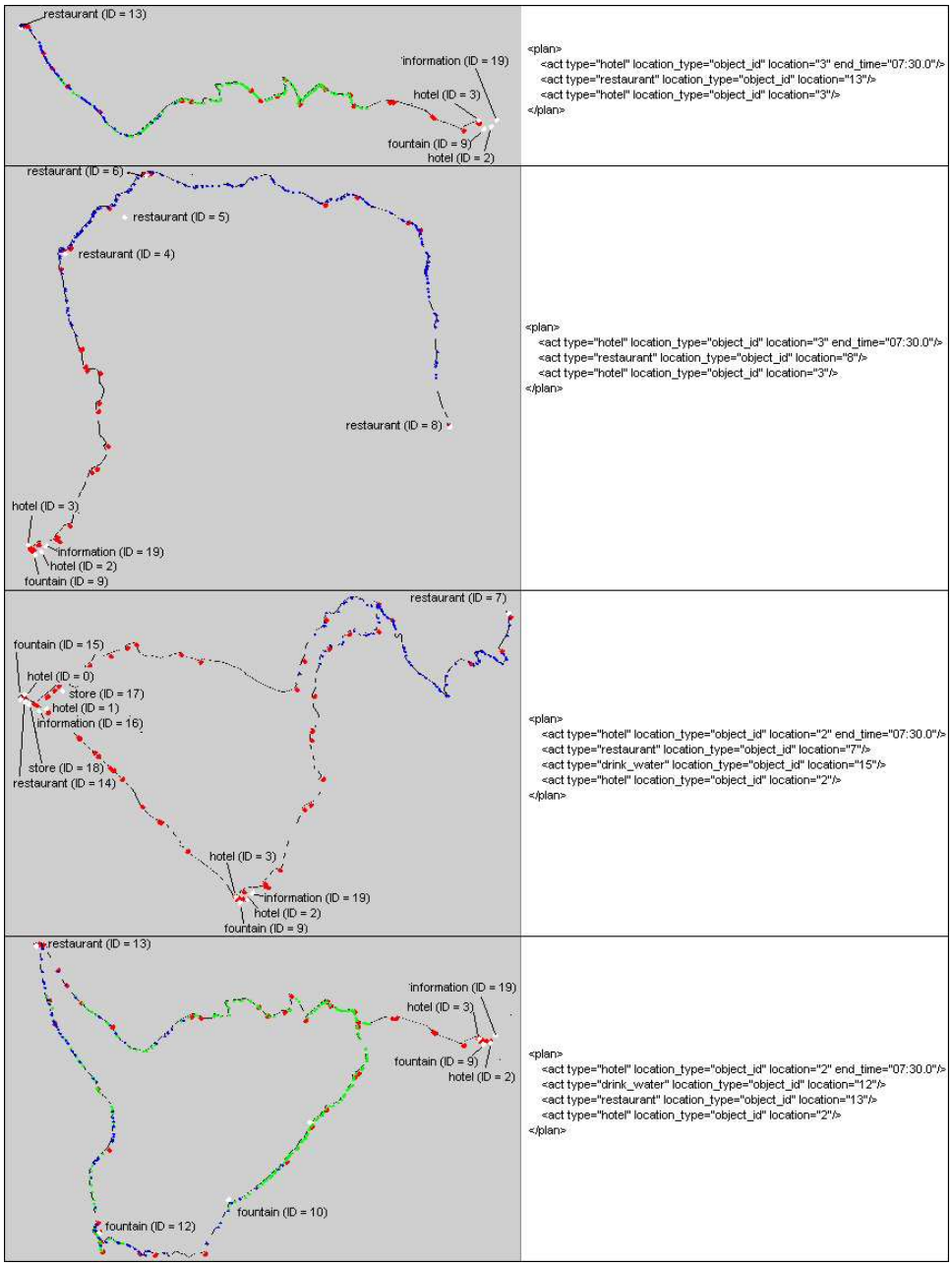


Abbildung A.3: Routen 7 bis 10. Links sind die Daten, welche der Agent wahrnimmt in kartographischer Form zu sehen. Die weissen beschrifteten Punkte sind Objekte die der Agent sehen kann. Die Roten Punkte sind Verzweigungen des Strassennetzwerks. Die grünen und blauen Punkte, stehen für Orte wo der Agent schöne Aussicht oder schöner Wald wahrnimmt. Die schwarzen Punkte verdeutlichen die eigentliche Route. Die Pläne sind rechts in XML Form gegeben.

Anmerkung: Falls die wahrgenommenen Daten bei der Ausführung der Routen zum initialisieren einer mentalen Karte verwendet werden ist Voraussetzung, dass sie mit einer Aktivität starten und enden. Die zweite Route in Abbildung A.1 erfüllt diese Bedingung nicht!

A.2 Grundeinstellungen der Mentalen Karte

In den verschiedenen Experimenten, die in Kapitel 4 beschrieben werden, wurden die in Tabelle A.1 angegebenen Grundeinstellungen, d.h. Parameter der mentalen Karte verwendet.

variable (XML name)	value	short description
STM_size	10	default size of the short time memory (STM)
LTM_hashtable_Size	500	initial default size of the hash table in the LTM
preferences	0.5	default preference value of an object
STM_SimilarityThreshold	0.6	default similarity threshold when comparing views
STM_Sensitivity	10	regulates the speed of the score decay in an object
STM_AdaptationSpeed	0.1	regulates how fast the score adapts to the contex of a local scene
STM_DecisionPressure	1.0	additional value to manipulate the object score
ROUTE_TimeDeviation	1.0	regulates the maximal deviation from the optimal execution time for an activity
ROUTE_BackwardDiscount	0.5	discounts the score when using a route objecs backwards
ROUTE_PedestrianVelocity	5	used to calculate a time estimation for the viewable objects when a rough distance is known
AGENT_communicativity	0.7	probability that the agent communicates with another agent at the same location

Tabelle A.1: Die Tabelle zeigt die Grundeinstellungen der Parameter, die in den Gleichungen und Algorithmen in den verschiedenen Kapiteln angegeben wurden. Die Parameter können via XML Definitionsdatei explizit angepasst werden.

Anhang B

Struktur der Simulation

Das Kapitel soll eine Übersicht über die einzelnen Softwarekomponenten der Simulation gegeben werden. Die Simulation lässt sich grundsätzlich in vier Teile aufspalten:

- Die Kommunikationsstruktur
- Die Agentenstruktur
- Die Struktur der mentalen Karte
- Die Struktur des Aktivitätenplaners

Die Kommunikationsstruktur verwaltet die verschiedenen Agenten, verteilt die Wahrnehmungsevents an die einzelnen Agenten und regelt die Kommunikation zwischen den Agenten. Die Agentenstruktur leitet Informationen der XML-Ereigniss-Strings an die korrekten 'Event-handler' weiter. Dort werden die Informationen intern verarbeitet und daraus die Datenstruktur aufgebaut. Die Struktur der mentalen Karte, verwaltet die interne Speicherstruktur. Sie ist auch für die Automatisierung des Agenten zuständig. Die Struktur des Aktivitätenplaners, beinhaltet die Generierung von Aktivitätsplänen. Sie stellt zugleich die Algorithmen zur Verfügung, die einen Aktivitätenplan mit räumlichem Wissen vervollständigt. Mit anderen Worten soll eine Route gefunden werden, auf der die verschiedenen Aktivitäten ausgeführt werden.

B.1 Die Kommunikationsstruktur

Der Agent muss in der Lage sein Ereignisse, d.h. die codierten XML-Strings, von der Simulation entgegenzunehmen und an den entsprechenden Agenten weiterzuleiten. Dazu wird der passende 'Handler' des Agenten für das Ereignis mit den empfangenen Informationen aufgerufen. Die ankommenden XML-Strings werden zuerst in einem Objekt 'EventStream-Buffer' zwischengespeichert bevor sie vom Objekt 'EventDispatcher', analysiert werden. Dieses besitzt einen XML-Parser. Die extrahierten Information werden via dem Objekt 'EventController' an den Agenten weitergeleitet. Das Objekt 'AgentController' verwaltet die einzelnen Agenten. Eine XML-Definitionsdatei (vgl. Untersektion B.5.2) definiert welche XML-Ereignisse der Agent interpretieren soll. Zu jedem Ereignisstyp muss im Objekt 'Agent' ein entsprechender 'Event-handler' vorhanden sein. Eine graphische Übersicht der einzelnen Objekte und deren Zusammenhänge ist in Abbildung B.1 gegeben. Ein Agent wird ebenfalls mit Hilfe einer XML-Definitionsdatei initialisiert (vgl. Untersektion B.5.1).

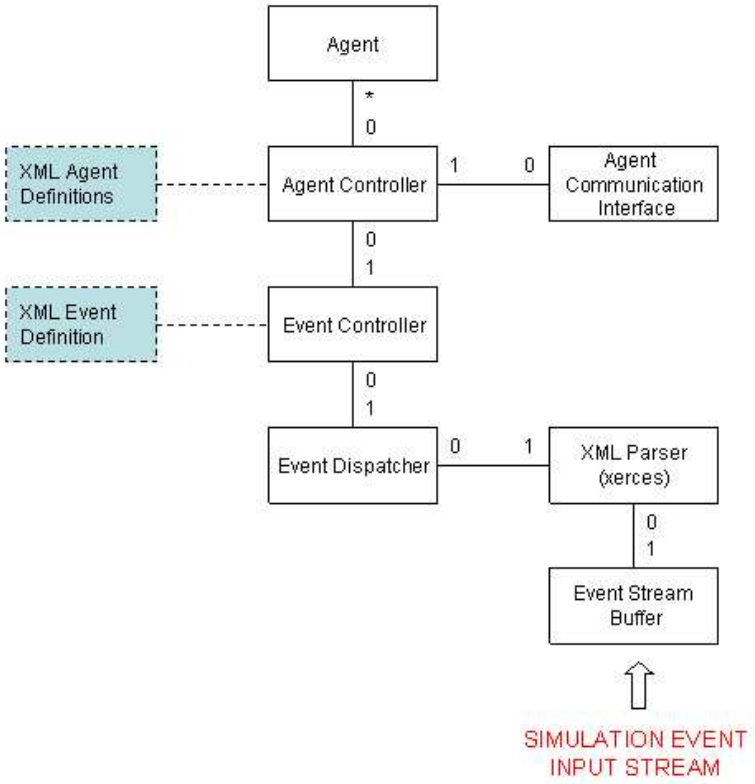


Abbildung B.1: Die Abbildung zeigt die Softwarestruktur, mit der ein Agent verwaltet und die Kommunikation mit der Simulation oder zwischen den Agenten realisiert wird. Neben den Objekten selbst sind die Kardinalitäten zwischen den einzelnen Objekten angegeben. Das Objekt 'Agent Controller' verwaltet und initialisiert die verschiedenen Agenten. Das Objekt 'AgentCommunicationInterface' bietet Kommunikationsmöglichkeiten zwischen den Agenten an. Die Event-Objekte regeln die Kommunikation mit der Simulation. Die ankommenden XML-Strings werden dabei zuerst gepuffert, bevor sie von einem XML-Parser interpretiert werden. Mit Hilfe der Objekte 'EventDispatcher' und 'EventController' wird die empfangene Information an die verschiedenen 'Event-handler' der Agenten weitergeleitet.

B.2 Die Agentenstruktur

Die Agentenstruktur besteht aus den 'Event-handler', die im Objekt 'Agent' definiert sind. Sie beinhaltet weiter ein Objekt 'MentalMap', welches die Zugangsschnittstelle zu den Informationen der mentalen Karte ist. Das Objekt ist gleichzeitig für die Automatisierung des Agenten verantwortlich. Das Objekt 'ViewController' baut die mentale Karte aus den ankommenden XML-Ereigniss-Informationen auf. Im Objekt 'GeneralAttributeKnowledge' sind die räumlichen Abhängigkeiten der einzelnen Attraktivitätstypen festgehalten. Jeder Agent besitzt weiter eine Tabelle, wo Identität, Koordinaten und Typ der wahrgenommenen Objekte assoziiert sind. Eine graphische Übersicht der einzelnen Objekte und deren Zusammenhänge ist in Abbildung B.2 gegeben.

B.3 Die Struktur der Mentalen Karte

Das Langzeitgedächtnis ist durch die mentale Karte gegeben. Das Objekt 'mentalMap' ist die Zugangsschnittstelle der mentalen Karte. Zusätzlich ist es dafür verantwortlich, dass bei nicht erwarteten Veränderungen der Umwelt entsprechend reagiert wird. Der Agent soll für

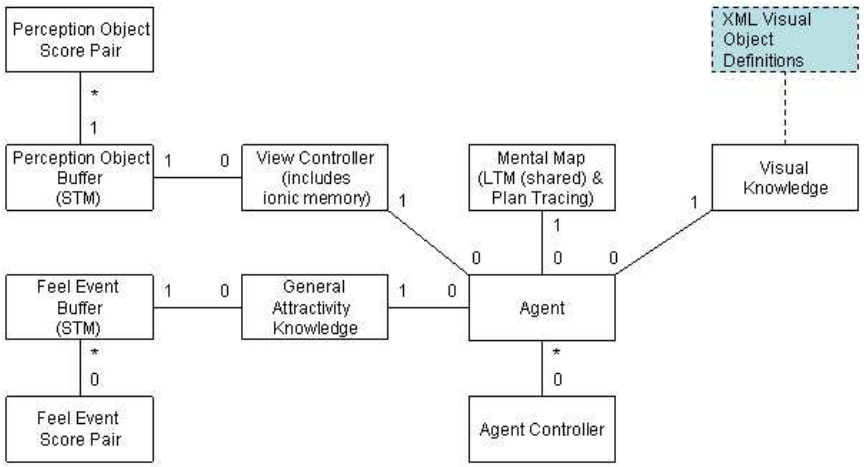


Abbildung B.2: Die Abbildung zeigt die Agentenstruktur. Das Objekt 'Agent' bietet die passenden 'Event-handler' zu den XML-Ereignis-Strings an. Die Informationen werden an die entsprechenden Objekte weitergeleitet, wo sie verarbeitet werden. Das Objekt 'View-Controller' baut dabei die mentale Karte auf. Die Automatisierung des Agenten sowie die Schnittstelle zur mentalen Karte sind im Objekt 'Mentale Karte' realisiert. In 'GeneralAttractivityKnowledge' werden die Abhängigkeiten der verschiedenen Attraktivitätstypen gelernt. Das Objekt 'Agent Interface' bietet dem Agenten gewisse Kommunikationsmöglichkeiten an.

jeden Tag einen neuen Aktivitätenplan generieren und diesen in seiner bekannten Umwelt ausführen. Das Objekt beinhaltet die entsprechende Automatisierung von Tagesabläufen.
Die mentale Karte basiert auf den einzelnen wahrgenommenen Objekten, den 'Perception-Objects'. Diese lassen sich zu einer visuellen Ansicht zusammenfügen, welche im Objekt 'View' wiedergegeben ist. Die einzelnen Ansichten sind durch sogennante 'action link'-Objekte verbunden. Orte die auch Verzweigungen sind, sind durch Routenobjekte ('Route'-Objekt) vernetzt. Mit den Routenobjekten werden die verschiedenen dort wahrgenommenen Attraktivitäten assoziiert. Die Attraktivität einer Route setzt sich aus verschiedenen Attraktivitätstypen zusammen. Die Attraktivitäten der verschiedenen Typen sind durch die Objekte 'AttractivityPattern' gegeben. Wird eine Route mehrmals begangen, so sollen die bereits gespeicherten Ansichten und damit verbundenen Objekte im Langzeitgedächtnis wiederverwendet und aktualisiert werden. Das statische Objekt 'ViewComparer' bietet dazu entsprechende Vergleichsalgorithmen für Ansichten an. Das Wissen der mentalen Karte wird vom Objekt 'ActivityPlaner' wiederverwendet. Ein Aktvitätenvorschlag wird dort mit räumlich und zeitlich passenden Informationen, d.h. Routenobjekten konkretisiert. Die mentale Karte agiert dabei als 'shared memory'. Der Aktivitätenvorschlag wird vom Objekt 'PlanGenerator' zufällig erzeugt. Sobald das Objekt 'MentalMap' das Ende des ausgeführten Aktivitätenplans detektiert wird, wird ein neuer Vorschlag für den nächsten Tag generiert. Der Agent ist automatisiert. Eine graphische Übersicht der einzelnen Objekte und deren Zusammenhänge ist in Abbildung B.3 gegeben.

B.4 Die Struktur des Aktvitätenplaners

Im Zentrum der Struktur des Aktivitätenplaners steht das Objekt 'ActivityPlaner'. Es probiert die verschiedenen Objekte, die für eine bestimmte Aktivität des vorgeschlagenen Aktivitätenplans verfügbar sind. Das Objekt 'Router' prüft die zeitliche und räumliche Verfügbarkeit anhand der mentalen Karte. Dort sind zwei entsprechende Algorithmen, Dijkstra und A* implementiert. Die Information im Objekt 'TypeKnowledge' wird verwendet, um die Aktivitätenvorschläge mit den Objekten in der mentalen Karte zu assoziieren. Wird ei-

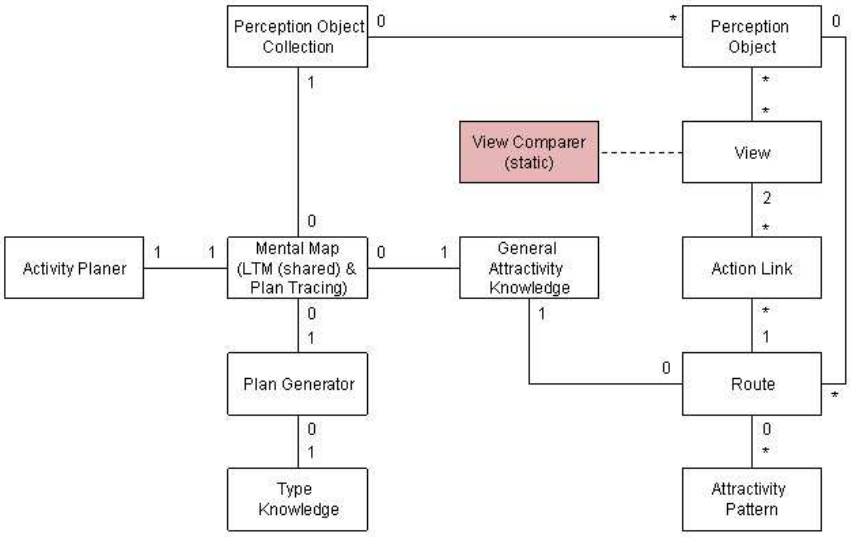


Abbildung B.3: Das Objekt 'MentalMap' repräsentiert einerseits eine Zugangsschnittstelle zur mentalen Karte. Anderseits sorgt sie gleichzeitig dafür, dass der Agent jeden Tag einen neuen Aktivitätenplan ausführt oder diesen bei unerwarteten Ereignissen anpasst. Die Basis der mentalen Karte sind die Objekte 'PerceptionObject', welche im Objekt 'PerceptionObjectCollection' abgelegt werden. Die vom Agenten wahrgenommenen Objekte, werden zu Ansichten, den view'-Objekten zusammengefügt. Das Objekt 'ViewComparer' bietet Methoden mit denen sich Ansichten vergleichen lassen. Aufeinanderfolgende Ansichten sind durch ein Objekt 'ActionLink' verbunden. Ein Objekt 'Route' beinhaltet die nacheinander wahrgenommenen 'ActionLink' Objekte zwischen zwei Verzweigungspunkten des Strassennetzwerks. Routenobjekte werden durch die assoziierten Objekte 'attractivityPattern' bewertet. Die gegenseitigen Abhängigkeiten der Attraktivitätstypen ist in Objekt 'GereralAttracitivityKnowledge' gegeben. Die Objekte 'ActivityPlaner' und 'PlanGenerator' generieren Tagesrouten für den Agenten aufgrund des in der mentalen Karte gespeicherten Wissens.

ne Route gefunden, so wird diese in dem Objekt 'RouteTable' Zwischengespeichert. Der Agent sucht nach weiteren möglichen Routen. Eine Route beginnt immer als ein Aktivitätenvorschlag, der vom Objekt 'PlanGenerator' erzeugt wird. Die Informationen zu einer Aktivität werden im Objekt 'Activity' gesammelt. Das Objekt 'Plan' fasst die Aktivitäten zusammen, die für eine Tagesplanung nötig sind. Der Suchalgorithmus vervollständigt die 'Activity' Objekte mit Wissen aus der mentalen Karte. Das Resultat ist eine Route. An den verschiedenen Orten werden Aktivitäten ausgeführt. Ein vervollständigter Plan kann entweder direkt an die Simulation weitergeschickt werden oder der Agent kann zusätzlich einen anderen Agenten fragen, ob er eine bessere Lösung hat. Die Kommunikation geschieht via das Objekt 'AgentCommunicationInterface'. Eine graphische Übersicht der einzelnen Objekte und deren Zusammenhänge ist in Abbildung B.4 gegeben.

B.5 XML-Defintions Dateien

Die Simulation wird via die Main-Klasse 'StartMapSim.java' gestartet. Bevor diese aufgerufen werden kann muss die Simulation via verschiedene XML Definitionsdateien initialisiert werden. In den folgenden Untersektionen ist eine Übersicht gegeben.

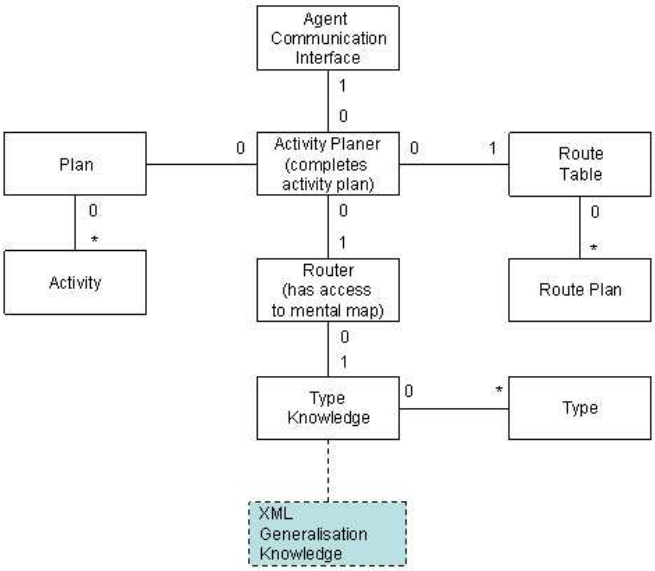


Abbildung B.4: Das Objekt 'ActivityPlaner' sucht in der mentalen Karte via das Objekt 'MentalMap' nach Informationen, um die 'Activity'-Objekte eine Aktivitätenplans mit räumlichen Informationen zu vervollständigen. Die 'Activity'-Objekte werden im Objekt 'Plan' verwaltet. Die gefundenen Vorschläge werden im Objekt 'RouteTable' zwischengespeichert. Sobald der Agent sein ganzes räumliches Wissen abgesucht hat, kann er zusätzlich noch einen anderen Agenten fragen. Dies geschieht via das Objekt 'AgentCommunicationInterface'.

B.5.1 XML Agenten Definition

In Abbildung B.5 ist ein Beispiel für eine Agentendefinition gegeben. Damit ein Agent in der Simulation erzeugt wird muss mindestens das Identitätsattribut angegeben werden. Eine Übersicht der Attribute ist in Appendix A.2 gegeben. Für Attribute die nicht explizit angegeben sind, wird der Basiswert verwendet.

B.5.2 Eventdefinition

Ein Beispiel für eine Eventdefinition ist in Abbildung B.6 gegeben. Für jedes Event, das der Agent empfängt und verarbeiten soll, muss ein 'Event handler' geschrieben werden. Dort werden die Attributinformationen weiterverarbeitet.

B.5.3 Visuelle Informationen

In einer Tabelle werden Typinformationen, Koordinaten und Identität eines Objektes assoziiert. Die Tabelle wird während der Simulationsinitialisierung aufgebaut. Darin enthalten sind nur die visuell wahrnehmbaren Objekte. In Abbildung B.7 ist eine entsprechende Definitionsdatei für die entsprechenden Objekte gegeben. Die Tabelle wird während der Simulationausführung automatisch mit Routenverzweigungsobjekten erweitert.

B.5.4 Generalisationswissen

In Abbildung B.8 ist ein Beispiel für die Definition einer Generalisierungshierarchie gegeben. Die Generalisationshierarchie assoziiert die Aktvitäten mit Objekttypen. Sie fasst zusätzlich Gruppen von Objekttypen in einem Obertyp zusammen.

```
<agents>
  <agent id = "1">

    <!-- agent information preferences -->
    <preferences>
      <!-- view preferences -->
      <view_preferences>
        <preference type = "restaurant" value = "0.5" />
        <preference type = "hotel" value = "0.5" />
        <preference type = "information" value = "0.5" />
        <preference type = "fountain" value = "0.5" />
        <preference type = "house" value = "0.5" />
        <preference type = "store" value = "0.5" />
        <preference type = "forest" value = "0.5" />
      </view_preferences>

      <!-- feel preferences -->
      <feel_preferences>
        <preference type = "nice_view" value ="0.5" />
        <preference type = "nice_forest" value ="0.5" />
        <preference type = "pedpressure" value ="0.0" />
        <preference type = "sunny" value ="1.0" />
      </feel_preferences>
    </preferences>

    <!-- initialisation data -->
    <initial_data_runs>
      <run path="/home/dkistler/src/dkistler/temp/runs/samplePlan3.xml" />
    </initial_data_runs>

    <!-- others (if not defined, the default value will be used) -->
    <STM_Sensitivity value="10.0"/>

  </agent>
</agents>
```

Abbildung B.5: Die Abbildung zeigt eine XML-Definitionsdatei für einen Agenten. Es kann eine Liste von Agentendefinitionen angegeben werden. Für jede Definition wird ein Agent initialisiert.

```
<event_types>

  <!-- perception events -->
  <event_type type="position"/>
  <event_type type="view"/>
  <event_type type="location"/>

  <event_type type="nice_forest"/>
  <event_type type="nice_view"/>
  <event_type type="pedpressure"/>
  <event_type type="sunny"/>

</event_types>
```

Abbildung B.6: Die Abbildung zeigt die XML Definitionsdatei für die Eventtypen, die der Agent verarbeiten können muss. Für jeden Eventtypen muss zusätzlich ein 'Event-handler' implementiert werden.

```
<visknowledge>
    <point ID="0" ... objecttype="hotel" ... x="588444.997878" y="150258.012133" ... />
    ...
    <point ID="4" ... objecttype="restaurant" ... x="590396.582006" y="149213.914625" ... />
    ...
    <point ID="9" ...  objecttype="fountain" ... x="590130.678669" y="151885.1454" ... />
    ...
    <point ID="11" ... objecttype="restaurant" ... x="586179.94055" y="154295.351798" ... />
    ...
    <point ID="17" ...  objecttype="store" ... x="588744.06722" y="150231.364067" ... />
    ...
    <point ID="19" ... objecttype="information" ... x="590228.107734" y="151812.615899" ... />
    ...
    <polygon ID="20" ... objecttype="forest" >
        <coord id="1" x="588820" y="150350"/>
        <coord id="2" x="588820" y="150450"/>
        <coord id="3" x="588920" y="150350"/>
        <coord id="4" x="588920" y="150450"/>
    </polygon>
    ...
    <polygon ID="329" ... objecttype="forest" ... >
        <coord x="588530.5" y="153114" />
        <coord x="588538.7" y="153115.3" />
        <coord x="588542.3" y="153118.2" />
        <coord x="588552.3" y="153136" />
        <coord x="588551.3" y="153157.8" />
        <coord x="588548.2" y="153166.4" />
        <coord x="588545.6" y="153167.7" />
        <coord x="588542.5" y="153163.1" />
        <coord x="588540.6" y="153149.3" />
        <coord x="588538.5" y="153145" />
        <coord x="588519.4" y="153127.6" />
        <coord x="588517.1" y="153122.6" />
        <coord x="588517.7" y="153119" />
        <coord x="588521.1" y="153116.7" />
        <coord x="588530.5" y="153114" />
    </polygon>
</visknowledge>
```

Abbildung B.7: Die Abbildung zeigt einen Ausschnitt aus den XML-Definitionen für visuelle Dateien. Aus Platzgründen werden nur die visuellen Attribute angezeigt, die tatschlich von der mentalen Karten Simulation verwendet werden.

```
<world>
    <still>
        <stillactivities>
            <rest_and_sleep>
                <hotel/>
            </rest_and_sleep>
            <drink_water>
                <fountain/>
            </drink_water>
            <buy_drink>
                <store/>
            </buy_drink>
            <buy_provisions>
                <store/>
                <shopping_center/>
            </buy_provisions>
            <get_local_information>
                <information/>
            </get_local_information>
            <eat>
                <hotel/>
                <restaurant/>
            </eat>
            <sightseeing>
                <castle/>
            </sightseeing>
        </stillactivities>
        <object>
            <traversable>
                <node/>
                <waypoint/>
                <building>
                    <house>
                        <shopping_center/>
                        <restaurant/>
                        <store/>
                        <hotel/>
                        <information/>
                        <castle/>
                            <fountain/>
                    </house>
                </building>
                <route>
                    <street/>
                </route>
            </traversable>
            <nontraversable>
                <forest/>
             </nontraversable>
        </object>
    </still>
    <move>
        <moveactivities>
            <hike/>
        </moveactivities>
    </move>
</world>
```

Abbildung B.8: Die Abbildung zeigt die vollständige Generalisationshierarchie, wie sie in der Simulation verwendet wurde.

Literaturverzeichnis

[1] Rubin D.C. Wenzel A.E. On the form of forgetting. *Psychological Science*, 2:409–415, 1991.

[2] Needleman S.B. Wunsch C.D. A general method applicable to the search for similarities in the amino-acidsequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

[3] Kahneman D. *Attention and Effort*. Prentice-Hall, EnglewoodCliffs New Jersey, 1973.

[4] Wixted T.J. Ebbesen E.B. One hundert years of forgetting: A quantitative description of retention. *Psychology Review*, 103:734–760, 1996.

[5] Gloor Ch. et al. A pedestrian simulation for very large scale applications. *A Koch und P Mandl(Hrsg.): Multi-Agenten-Systeme in der Geographie. Klagenfurt.*, Heft 23, 2003.

[6] Jensen F.V. Bayesian networks basics. *AISB Quarterly*, 94:9–22, 1996.

[7] Mandler G. Recognizing: The judgement of previous occurence. *Psychological Review*, 87:252–271, 1980.

[8] Hirtle S.C. Joinides J. Evidence of hierarchies in cognitive maps. *Memory and Cognition*, 13:208–217, 1985.

[9] Theeuwes J. Visual selective attention: A theoretical analysis. *Acta Psychologica*, 83:93–154, 1993.

[10] Benjamin Kuipers. The map in the head metaphor. *Environment and Behavior*, 14:202–220, 1982.

[11] Rabiner L. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[12] Yarbus A. L. *Eye movements during perception of complex objects in L. A. Riggs ed. 'Eye Movements and Vision'*, chapter chapter VII, pages 171–196. Plenum Press, New York, 1 edition, 1967.

[13] Maio D. Rizzi S. Map learning and clustering in autonomous systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1286–1297, 1993.

[14] Maio D. Rizzi S. *Layered knowledge architecture for navigation-oriented environment representation*. University of Bolognia, technical report cioc-c.n.r., n.18 edition, 1996.

[15] Timpf S. *Hierarchical structure in map series*. PhD thesis, Technical University Wien, 1998.

[16] Riesenhuber M. Poggio T. T. models of object recognition. *Nature Neuroscience*, 3:1199–1204, 2000.

[17] Lee T.R. Perceived distance as a function of direction in the city. *Environment and Behavior*, 2:40–51, 1970.