Dokumentation Bufferlo

1. Zielsetzung

Ziel des Bufferlo-Projekts ist es, ein Programm zur Berechnung der Geltungsbereiche von Raumausdrücken an Objekten zu entwickeln. Das Projekt soll als allgemeine Grundlage für Programme dienen, die mit Raumausdrücken arbeiten und dabei den umgekehrten Weg der Umwandlung von räumlichen Verhältnissen in Sprache gehen müssen. Als Anwendungsgebiete wären z.B. Navigations- oder Bildbeschreibungssysteme denkbar.

2. Konzept

Die geometrischen Objekte, für die die Geltungsbereiche der Raumausdrücke berechnet werden sollen, sollen durch zweidimensionale Polygone repräsentiert werden, ebenso die errechneten Geltungsbereiche (Buffer) selbst. Überlegungen, die Berechnungen auch an dreidimensionalen Objekten durchzuführen, wurden schon nach kurzem wegen der unabsehbaren Komplexität dieser Aufgabe fallengelassen. Dreidimensionalität ist auch für die anvisierten Anwendungsbereiche des Programms keine Notwendigkeit.

Nach der Auswahl eines Raumausdrucks und eines oder mehrerer Objekte – eventuell durch die Eingabe einer natürlichsprachlichen Präpositionalphrase – soll der entsprechende Buffer berechnet werden. Zur besseren Handhabbarkeit ist eine grafische Benutzeroberfläche vorgesehen, auf der die geometrischen Objekte und die berechneten Buffer dargestellt werden. Eine Anzahl geometrischer Objekte soll sich auf einer sogenannten Karte zusammenfassen lassen. Das bedeutet jedoch nicht, dass nur an die Verarbeitung von Stadtplänen und Landkarten gedacht wurde – auch zum Beispiel die Objekte auf einer Schreibtischoberfläche ließen sich auf einer solchen Karte repräsentieren. Die Kartendefinition soll zwei verschiedene Sichtweisen erlauben: Grundriss und Aufriss.

Wichtigstes Ziel ist es, die Berechnung der Buffer weitestgehend parametergesteuert zu gestalten. Die Aufnahme neuer Präpositionen soll möglich sein, ohne den Programmcode ändern zu müssen – lediglich durch das Eintragen der Parameter in eine Präpositionsdatenbank. Wird dieses Ziel erreicht, sollte auch die Übertragung der Präpositionsdatenbank in andere Sprachen leicht möglich sein.

3. Analyse der Raumausdrücke

3.1. Allgemeines

Wenn im Rahmen der Bufferlo-Dokumentation von Präpositionen gesprochen wird, sind damit stets Raumausdrücke gemeint – also Ausdrücke, die ein räumliches Verhältnis zwischen Objekten beschreiben. Zum größten Teil sind dies Präpositionen im herkömmlichen, grammatischen Sinn, aber auch Ausdrücke wie zum Beispiel „im Zentrum“ sind Raumausdrücke, wenn auch keine Präposition nach der strengen Wortdefinition.

Die Projektarbeit begann damit, die Raumausdrücke des Deutschen zu sammeln – allerdings ohne Vollständigkeitsanspruch – und zu analysieren. Ziel der Analyse war es, die Präpositionen nach der Art der Flächen, die sie erzeugen, in Gruppen einzuteilen und die Parameter zu bestimmen, die das Aussehen dieser Flächen beeinflussen.

Folgende 37 Raumausdrücke wurden untersucht:

> an, an...entlang, auf (+Dat), auf (+ Akk), auf...zu, aus..heraus, außerhalb, bei, durch, entlang, gegenüber, herauf, herunter, hinauf, hinunter, hinter, im Zentrum, in (+Akk.), in (+Dat), in der Mitte, innerhalb, links, nahe, neben, nördlich, oberhalb, östlich, rechts, südlich, über (+ Akk), über (+ Dat), um...herum, unter, unterhalb, vor, westlich, zwischen

3.2. Statische und dynamische Raumausdrücke

Raumausdrücke lassen sich grundsätzlich in zwei Gruppen aufteilen: in statische und dynamische. Statische Raumausdrücke treffen zu, wenn sich das von ihnen zu lokalisierende Objekt (LO) im Geltungsbereich dieser Präposition am Referenzobjekt (RO) befindet: „Der Mülleimer (LO) steht neben dem Haus (RO)“. Für statische Raumausdrücke läßt sich also, in Abhängigkeit von der Beschaffenheit des Referenzobjekts, eine Fläche angeben, die den Geltungsbereich des jeweiligen Raumausdrucks beschreibt.

Dynamische Raumausdrücke treffen zu, wenn das LO im Verhältnis zum RO eine bestimme Bewegung ausführt: „Das Auto (LO) fuhr an der Mauer (RO) entlang“. Zumindest für das genannte Beispiel kann man wahrscheinlich eine Fläche angeben, die den Geltungsbereich der Präposition beschreibt. Allerdings genügt es nicht, wenn sich das LO innerhalb dieser Fläche befindet – es muss statt dessen innerhalb dieser Fläche eine in Bezug auf Richtung und Dauer nicht beliebige Bewegung vollführen. Hinzu kommt noch, dass die Geltungsbereiche anderer dynamischer Raumausdrücke sich kaum durch Flächen beschreiben lassen. Man denke hierbei zum Beispiel an Sätze wie „Ich gehe in das Haus“. Hierbei ist es nur wichtig, dass ich vor der Bewegung außerhalb, und nach der Bewegung innerhalb des Hauses bin. Selbst dass ich höchstwahrscheinlich durch eine Tür gehe, ist eigentlich nicht relevant, wie man sieht, wenn man „Tür“ durch „Kreis" ersetzt.

Wir beschlossen, uns vorerst auf die statischen Raumausdrücke zu konzentrieren. Auf die Schwierigkeiten bei der Analyse und Darstellung dynamischer Raumausdrücke wurde bereits im letzten Absatz hingewiesen. Auch nach längeren Überlegungen blieb unklar, wie zum einen die notwendige Bewegung des LO innerhalb dieser Flächen kenntlich gemacht werden soll, und zum anderen inwiefern die Beschreibung durch Flächenangaben überhaupt adäquat sein können.

3.3. Sichtweisen statischer Raumausdrücke

Die Geltungsbereiche statischer Raumausdrücke sind zumeist sichtweisenabhängig. Der Buffer kann einmal aus der Sicht des Referenzobjekts heraus erstellt werden: diese Sichtweise heißt intrinsisch. „Links des Hauses“ aus intrinsischer Sicht zum Beispiel entsteht auf der Seite des Hauses, auf die der linke Arm einer Person zeigen würde, die in der Tür dieses Hauses steht und nach draußen sieht. Das bedeutet, dass Buffer aus intrinsischer Sicht nur

dann definiert sind, wenn das Referenzobjekt gerichtet ist und man also eine Vorderseite angeben kann.

Die zweite Sichtweise statischer Raumausdrücke heißt deiktisch. Hier werden sowohl RO als auch LO von einer dritten Instanz, einem sogenannten Betrachter, aus gesehen. Wenn dieser Betrachter vor dem eben erwähnten Haus stünde (also vor dessen Vorderseite), so wäre für ihn das, was aus intrinsischer Sicht „links des Hauses“ ist, „rechts des Hauses“. Stünde der Betrachter auf der Rückseite des Hauses und wollte eine Regentonne lokalisieren, die dort an der Hauswand steht, so könnte er dies tun, in dem er sagte: „Die Regentonne steht vor dem Haus“. Gibt er an, sie befinde sich hinter dem Haus, so bedeutet das, dass er sich in die Geometrie des Hauses eingedacht und aus intrinsischer Sicht gesprochen hat.

Aus den beiden Sichtweisen intrinisch und deiktisch ergeben sich zwei neue Konzepte für das Bufferlo-Projekt. Erstens soll es möglich sein, gerichtete geometrische Objekte zu erstellen. Durch einen Richtungsvektor im Objekt soll man angeben können, wo sich dessen Vorderseite befindet. Zweitens soll man bei der Berechnung des Buffers zusätzlich die Koordinaten eines Betrachters angeben können, von dem aus der Geltungsbereich der Präposition gesehen wird. Auf der GUI sollen sowohl die Richtungsvektoren der Objekte dargestellt als auch das Setzen eines Betrachters ermöglicht werden. Die momentan implementierte Datenstrukur für den Betrachter sieht außerdem einen Namen und eine ID für den Betrachter vor und ermöglicht auch die Festlegung einer Blickrichtung. Name und ID wurden aufgenommen, damit man die Implentierung der GUI leicht so ändern kann, dass man zwischen verschiedenen Betrachtern hin- und herschalten kann, ohne jedesmal die Betrachterdaten neu eingeben muß (zur Bedienung der GUI siehe → Die GUI). Die Blickrichtung des Betrachter ist bei den von uns untersuchten Raumausdrücken nicht wichtig, da sich der Betrachter im Sprachgebrauch immer vorstellt, er blickt in Richtung des ROs. Daher ist die Verwendung einer Betrachterrichtung optional.

Es gibt jedoch auch Raumausdrücke, die innerhalb eigener Bezugssysteme operieren. Das sind zu erst einmal die vier Himmelsrichtungen, deren Geltungsbereiche unabhängig von einer bestimmten Sichtweise definiert sind. Um sie zu erfassen, haben wir einen weiteren Sichtweisentyp definiert: geo-global. Aber auch „über“ und „unter“ bedeuten sowohl aus intrinsischer als auch deiktischer Sicht dasselbe – ihre Definition scheint von der Richtung der Gravitation abzuhängen.

3.4. Modifikatoren

Bei der Analyse der Raumausdrücken fielen uns Wörter auf, die keine selbstständigen Raumausdrücke darstellen, aber deren Geltungsbereiche modifizieren. Dementsprechend haben wir sie Modifikatoren genannt. Ein Beispiel: in der Präpositionalphrase „oben im Haus“ ist „oben“ Modifikator; er verkleinert den Geltungsbereich von „in“.

Die gefundenen Modifikatoren, sortiert nach den Geometrieachsen, auf denen die Änderungen wirksam werden:

- auf der horizontalen Achse: *links, rechts*
- auf der vertikalen Achse: *oben, unten*
- auf der Tiefenachse: *hinten, vorne*
- außerdem: *schräg*

„Links“ und „rechts“ lassen sich dabei nicht nur als Modifikatoren, sondern auch als eigenständige Präpositionen verwenden.

Auch für Modifikatoren sollen Parameter gefunden werden, die ihren Einfluß auf die Geltungsbereiche von Raumausdrücken fassbar machen. Wie bei den Präpositionen ist es das Ziel, zur Aufnahme neuer oder Übersetzung bekannter Modifikatoren nicht den Programmcode der Berechnungsalgorithmen, sondern lediglich Parametersätze in einer Datenbank editieren zu müssen.

3.5. Genauigkeit von Geltungsbereichen

Wie bereits gesagt, sollen sowohl die geometrischen Objekte als auch die dafür berechneten Buffer durch Polygone repräsentiert werden. Dabei entsteht folgendes Problem: ein Polygon hat klar definierte Ränder; der Geltungsbereich von Raumausdrücken jedoch nicht. Sicher gibt es eine Kernzone, in der der Raumausdruck auf jeden Fall gültig ist, aber es ist für Menschen schwer zu entscheiden und auch individuell verschieden, in welcher Entfernung vom Referenzobjekt der Geltungsbereich einer Präposition endet. Auch scheint es so zu sein, dass sich die Geltungszonen „benachbarter“ Präpositionen überlappen; in Grenzbereichen kann sich ein Objekt sowohl neben als auch hinter einem anderen befinden. In jedem Fall ist die Größe des Buffers abhängig von der Größe des Referenzobjekts: „vor der Ameise“ erzeugt einen wesentlich kleineren Buffer als „vor dem Elefanten“.

Aus diesen Überlegungen ergeben sich neue Parameter für die Repräsentation der Raumausdrücke. Die Größe des Buffers soll als Prozentwert der Größe des Referenzobjekts angegeben werden. An den linken und rechten Rändern eines Buffers sollen sich Winkel anlegen lassen, um den Geltungsbereich zu vergrößern und das „Überlappen“ von Buffern zu repräsentieren.

Die graduelle Abnahme der Geltungswahrscheinlichkeit von Raumausdrücken mit wachsender Entfernung vom RO könnte durch eine dementsprechende Schraffur des Buffers kenntlich gemacht werden. Es müssten Koordinaten, Form (z.B. Punkt oder Linie) und Größe einer Kernzone angegeben werden, dazu die prozentuale Abnahme der Geltungswahrscheinlichkeit pro Entfernungsschritt. Die Größe eines Entfernungsschritts stellt vermutlich einen weiteren Parameter dar. Leider ist es uns nicht mehr gelungen, die Bufferschraffur zu implementieren.

3.6. Aufteilung in Flächentypen

Schlußendlich haben wir die untersuchten Raumausdrücke nach der Form der Flächen, die sie erzeugen, in Gruppen eingeteilt. Diese Gruppen bzw. Typen haben wir mit Großbuchstaben bezeichnet. Sie sind folgendermaßen definiert:

A: Fläche(n) an einem Objekt (z.B. *vor*)
B: Fläche zwischen mehreren Objekten (z.B. *zwischen*)
C: Ringfläche um ein Objekt: (z.B. *bei*)
D: Fläche innerhalb der Objektgrenzen (z.B. *in*)
E: Linie (z.B. *auf* in der Seitenansicht)

Unsere Hoffnung ist es, dass pro Flächentyp lediglich ein Berechnungsalgorithmus zum Erzeugen der Buffer nötig ist. Alle Raumausdrücke eines Typs ließen sich dann lediglich durch die Angabe unterschiedlicher Werte der zur Berechnung nötigen Parameter definieren.

Es ist möglich, zu einem Raumausdruck mehrere Parametersätze zu definieren. So erzeugt zum Beispiel „neben“ zwei Flächen und hat dementsprechend auch zwei Parametersätze. Denkbar ist auch, dass sich die von einer Präposition erzeugte Fläche aus mehreren Flächen verschiedener Flächentypen zusammensetzt. „Durch“ zum Beispiel könnte man mit drei Flächen der Typen A, D, und wieder A definieren: eine Fläche vor dem Eingang, eine Fläche innerhalb und eine dritte vor dem Ausgang des Objekts. Werden alle drei Flächen beschritten, hat sich ein Objekt „durch“ das andere bewegt.

4. Programmstruktur

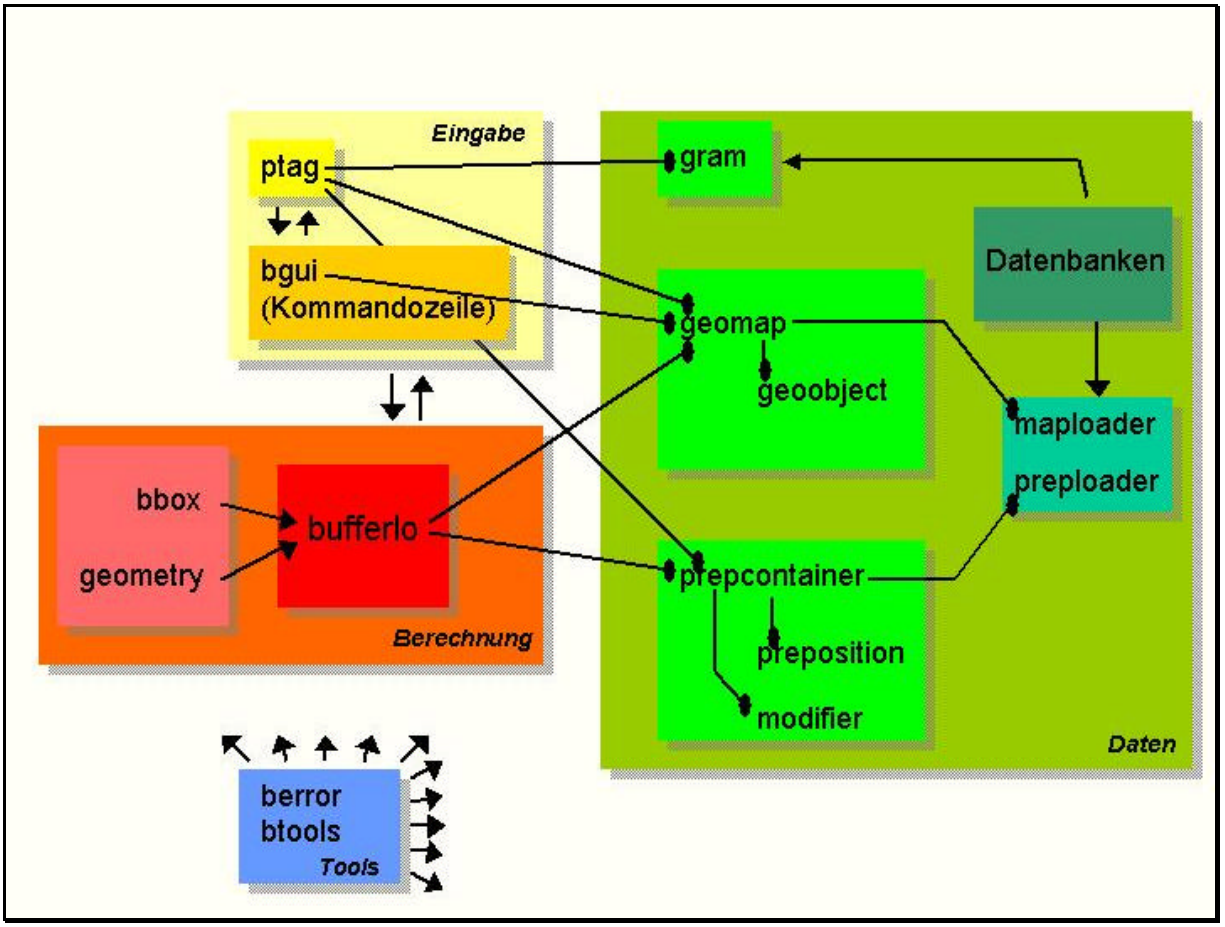

Abbildung 1 - Programmstruktur

Abbildung 1 beschreibt die Struktur und die Abhängigkeiten der verschiedenen Programm-module. Sie ist in vier unterschiedlich gefärbte Bereiche gegliedert, die verschiedene Aufgabenbereiche symbolisieren:

Eingabe – gelb
Berechnung – rot
Daten – grün
Tools – blau

Es folgt eine kurze Übersicht über die Programmmodule in den verschiedenen Bereichen. Für eine genauere Beschreibung der einzelnen Module siehe das gleichnamige Kapitel und die aus den Kommentaren im Quellcode automatisch generierte Dokumentation.

4.1. Eingabe

Das Modul `bgui` beeinhaltet die graphische Benutzeroberfläche und ist demnach nicht nur eine Eingabe-, sondern natürlich auch eine Ausgabekomponente. Die Kommandozeile ist hier in Klammern aufgeführt, um anzudeuten, dass es sich nicht um ein eigenständiges Modul handelt. Vielmehr läßt sich das Modul `bufferlo` auch durch Kommandozeilenparameter steuern. Das Modul `ptag` schließlich beeinhaltet einen Parser für Präpositionalphrasen.

4.2. Berechnung

`bufferlo` ist die zentrale Komponente des Programms; hier werden die Geltungsbereiche der Raumausdrücke berechnet. `bbox` ist eine Sammlung von Klassen zum Erstellen sogenannter BoundingBoxes. Diese sind grundlegend bei der Berechnung der Buffer und werden im Kapitel Module unter Bufferlo erklärt.

Das Modul `geometry` enthält Klassen, die allgemeine geometrische Objekte (Punkt, Linie, Kreis, Polygon) beschreiben, sowie zahlreiche Funktionen, die geometrische Tests auf der Grundlage dieser Klassen ausführen: Abstände messen, Winkel messen und ziehen, Schnittpunkte bestimmen. Diese Bibliothek wurde uns (ein großes Dankeschön!!) von Titus von der Malsburg zur Verfügung gestellt, der sie für ein Projekt im Rahmen eines Seminars über Sprachgenerierung entwickelt hat.

4.3. Daten

Die Module `geoobject`, `preposition` und `modifier` beinhalten Klassendefinitionen für geometrische Objekte, Präpositionen und Modifikatoren. `geomap` ist die Klassendefinition einer Karte (siehe → Konzept), in ihr können beliebig viele geometrische Objekte gespeichert werden. `prepcontainer` beinhaltet eine Klasse zur Verwaltung von Präpositionen und Modifikatoren.

Mit Hilfe der Module `maploader` und `preploader` können `geomap` und `prepcontainer` Daten von Karten und geometrischen Objekten bzw. Präpositionen und Modifikatoren aus externen Quellen laden – also zum Beispiel aus XML-Dateien. Für diese Aufgabe gibt es eigenständige Module, damit bei einem Wechsel des Quellentyps (etwa, wenn die Daten über das Internet gelesen werden sollen) lediglich diese Module ausgetauscht bzw. geändert werden müssen.

Das Modul `gram` schließlich beinhaltet die Definitionen von Wortklassen, die der Parser benötigt. Auch die Klassen, um die Definitionen dieser Wörter aus externen Quellen laden zu können, befinden sich hier. Sie sind jedoch ebenfalls von den Klassen, deren Daten sie laden, unabhängig und separat austauschbar.

Für die Definitionen der Datenbanken siehe bitte das Kapitel → Ressourcen.

4.4. Tools

Das Modul `btools` enthält einige Funktionen zur Listen- und Stringmanipulation. `berror` beinhaltet Klassendefinitionen für die selbstdefinierten Fehler, die bei der Programmausführung auftreten können.

5. Die Module im Detail

5.1. `bufferlo`

Das Modul `bufferlo` besteht aus einer in Python implementierten Klasse `Bufferlo` und einem Anweisungsteil für die Ausführung des Moduls als alleinstehendes Script.

5.1.1. Input

Das Modul `bufferlo` benötigt im ganzen 8 Parameter, um einen Buffer zu berechnen. Die ersten 6 Parameter sind obligatorisch:

1. Die ID(s) des oder der Objekte, für die ein Buffer berechnet werden soll. Im Großteil der Fälle wird dies eine einzelne ID sein. Nur für Präpositionen, die mehrere Referenzobjekte haben, wie z.B. „zwischen“, können hier mehrere Objekt-IDs übergeben werden.
2. Die Quelle, aus der die zur Berechnung notwendigen Objektdaten geholt werden sollen. Dies könnte eine Datei, oder auch eine Web-Adresse sein.
3. Die ID der Präposition, deren Geltungsbereich berechnet werden soll. Im Gegesantz zu 1. kann selbstverständlich nur eine einzelne ID angegeben werden.
4. Die Quelle, aus der die zur Berechnung notwendigen Parameter der Präposition geholt werden sollen. Dies könnte eine Datei, oder auch eine Web-Adresse sein.
5. Den Betrachter. Falls Werte für den Betrachter übergeben werden, findet die Berechnung unter Berücksichtigung dieser Werte und somit deiktisch statt, soweit die Präposition dies erlaubt. Falls keine Werte, bzw. `None`, für den Betrachter angegeben werden, findet die Berechnung je nach Präposition nach intrinsischer oder geo-globaler Sichtweise statt.
6. Die ID(s) des oder der Modifikatoren. Da Kombinationen wie *„rechts oben* an der Kirche“ im zweidimensionalen Rahmen unseres Projektes eher selten Sinn machen, wird auch hier im Großteil der Fälle eine einzelne ID übergeben werden. Möglich sind dennoch maximal zwei Modifikatoren. Eine Datenquelle wie in 2. und 4. muss nicht angegeben werden, da sich die Parameter der Modifikatoren in derselben Quelle befinden, wie die Parameter der Präpositionen.
   Anmerkung: Die Angabe einer Modifikator-ID ist zwar möglich, wird aber in der gegenwärtigen Version des Moduls ignoriert.

Die letzten beiden Parameter sind optional und werden bei fehlender Angabe auf `XML` gesetzt.

7. Die Angabe des Datenformats von 2. Bisher ist nur XML vorgesehen.
8. Die Angabe des Datenformats von 4. Bisher ist nur XML vorgesehen.

Die Übergabe der Parameter an das Modul kann auf zwei Wegen erfolgen. Zum einen ist es möglich, eine Objektinstanz der Klasse `Bufferlo` zu schaffen. Dieser Instanz müssen dann während der Instanziierung mindestens die ersten 6 Parameter übergeben werden (zur genauen Syntax siehe Quellecode `bufferlo.py`).

Der zweite Weg ist die Angabe der Parameter als Kommandozeilenparameter beim Aufruf des Moduls als alleinstehendes Script. Die Parameter werden einfach in o.g. Reihenfolge hinter den Script-Aufruf gesetzt, ohne spezielle Angabe von Switches in der Art von `-x`.

Dabei sind nur die ersten vier der oben genannten Parameter obligatorisch, weiterhin ist die Parameterangabe zum Betrachter aufgeteilt in eine Angabe zu den Betrachterkoordinaten (5.a.) und eine zur Blickrichtung des Betrachters (5.b.). Vom letzten, also 8., ausgehend, können die optionalen Parameter sukzessive weggelassen werden. Will man einzelne Parameter innerhalb dieser Reihe weglassen, so müssen an der Kommandozeile Leerstrings an den entsprechenden Stellen übergeben werden.

Beispiele:

```
python bufferlo.py 14 karten\stadt.xml 02 lingdata\praeps.xml
python bufferlo.py 14 karten\stadt.xml 02 lingdata\praeps.xml "450,50" "" 03
```

Im ersten Beispiel wird also für das Objekt mit der ID `14` aus der Datei `Karten\Stadt.xml` der Geltungsbereich der Präposition mit der ID `02` aus der Dateien `lingdata\praeps.xml` ausgerechnet. Da keine weiteren Parameter angegeben sind, wird zur Berechnung die intrinsische Sichtweise benutzt, der Buffer nicht weiter modifiziert und von im XML-Format vorliegenden Daten ausgegangen.

Das zweite Beispiel hingegen benutzt für die Berechnung eine deiktische Sichtweise mit einem Betrachter an den Koordinaten (`450/50`) und modifiziert den entstehenden Buffer entsprechend der Parameter des Modifikators mit der ID `03` (wieder aus der Datei `lingdata\praeps.xml`). *Anmerkung:* Die Angabe einer Modifikator-ID ist zwar möglich, wird aber in der gegenwärtigen Version des Moduls ignoriert.

5.1.2. Output

Das Ergebnis der Berechnungen wird in einem Datenattribut der Objektinstanz gespeichert, auf das – wie auf alle Attribute eines Python-Objektes – auch von außen (also öffentlich) zugegriffen werden kann.
In dieser Datenstruktur namens `buffer` werden folgende Ergebniskomponenten abgelegt:

1. Die Koordinatenangaben zu einem oder mehreren Buffern.
2. Die sichtweisenabhängige BoundingBox
3. Die sichtweisenunabhängige BoundingBox

zu 1.: Es ist möglich, dass eine einzelne Präposition einen Geltungsbereich generiert, der aus mehreren einzelnen Buffern besteht. So generiert zum Beispiel „neben“ jeweils eine Fläche links und eine Fläche rechts des Objektes. Die Parameter für beide Flächen

müssen einzeln bei der Präposition angegeben sein (siehe → Präpositionsdatenbank).

Die Informationen zu den einzelnen berechneten Buffern beinhalten zusätzlich zu den Koordinaten noch einen Namen, der im Moment immer „Buffer“ lautet, und eine ID, die im Moment immer „01“ lautet. Die verwendete Datenstruktur würde es zudem ermöglichen:

i. einen Richtungspfeil und somit einen gerichteten Vektor,
ii. semantische Informationen in Typ-Wert Paaren,
iii. im Buffer beinhaltete weitere Buffer und/oder
iv. grammatikalische Informationen zurückzugeben.

zu 2. und 3.: Zur Rolle der Boundingboxen siehe → Verarbeitung. Da sich auch im Falle mehrerer Bufferberechnungen zu einem Geltungsbereich aber weder das Referenzobjekt noch die Sichtweise auf dieses verändern, sind also sämtliche während der Berechnung angelegten Boundingboxen identisch. Daher werden nur die bei der letzten Berechnung angelegten Boundingboxen gespeichert.

Die Speicherung innerhalb des Moduls `bufferlo` erfolgt hauptsächlich, damit in die Boundingboxen in der GUI angezeigt werden können. Da die Berechnung der Boundingboxen durch ein eigenes Modul erfolgt, könnte man sie zwar auch in der GUI nochmals durchführen, was aber spätestens durch die aufwändigen Sichtweisenunterscheidungen äußerst ineffizient würde.

Genauere Angaben zur Syntax dieser Rückgaben finden sich in `bufferlo.py`, dem kommentierten Quellcode des Moduls.

5.1.3. Verarbeitung

Nachdem das/die durch die Inputparameter spezifizierte(n) Objekt(e) und die Präposition mit Hilfe der Module `geomap` und `prepcontainer` geladen wurden, wird für jeden bei der Präposition angegebenen Parametersatz (siehe → ...) ein Buffer berechnet.

Bei jeder einzelnen Berechnung wird dann nach den Informationen im jeweiligen Parametersatz unterschieden, zu welcher Flächenklasse (siehe → ...) der Buffer gehört. Da momentan nur die Berechnungen für den Typ A implentiert sind, erfolgt in allen andere Fällen eine Fehlermeldung, und nur wenn der zu berechnende Buffer vom Typ A ist, wird zur entsprechenden Berechnung gesprungen.

Im folgenden wird der Typ-A-Algorithmus genauer beschrieben. Wie weit sich diese Beschreibung mit der Beschreibung für die Algorithmen für die restlichen Typen decken wird, ist momentan nicht absehbar. Da das Konzept der Boundingbox, und speziell auch die von uns verwendete Typisierung, für die Berechnung essentiell ist, folgt zunächst eine Erklärung hierzu.

5.1.3.1. Bounding Box

Im Generellen ist eine Boundingbox (BB) ein Rechteck, das ein gegebenes Objekt so umschließt, dass jede der vier BB-Seiten von mindestens einem Punkt des Objektes berührt werden. Eine BB in diesem Sinn kann, aber muss nicht das kleinstmögliche einschließende

Rechteck sein. Im Bufferlo-Projekt unterscheiden wir zwischen vier verschiedenen BBs. Bei allen vieren ist genau festgelegt, wie die Seiten des Rechtecks durchnummeriert werden.

Die Default-BB ist parallel zur längsten Linie des Objektes ausgerichtet. Das bedeutet also, dass die 1. Linie parallel zur längsten Objektlinie ist und diese unter Umstände auch überdeckt. Die weiteren Linien werden im Uhrzeigersinn durchnummeriert. Wir haben diese BB „Default-BB“ genannt, weil es die BB ist, die für die meisten Menschen intuitiv am zugänglichsten, ist.

Die intrinische BB ist senkrecht zur Objektrichtung ausgerichtet. Das heißt also, dass die 1. Linie senkrecht zum Richtungsvektor steht und von diesem, wenn er entsprechend gestreckt würde, geschnitten würde. Die weiteren Linien werden gegen den Uhrzeigersinn durchnummeriert. *Anmerkung:* Eine eigentlich unerwünschte Folge dieser Definition ist, dass der eventuell beim Objekt angegebene Richtungsvektor (siehe → ...) das Objekt nicht so schneiden darf, dass er „von außen in das Objekt hinein geht.“

Die deiktische BB ist im Bezug auf den Betrachter ausgerichtet. Die 1. BB-Linie muss dafür senkrecht auf einer imaginären Linie, die vom Objektmittelpunkt zum Betrachter geht, stehen. Die weiteren Linien werden gegen den Uhrzeigersinn durchnummeriert. Diese BB kann selbstverständlich nicht erstellt werden, wenn sich der Betrachter innerhalb des betroffenen Objektes befindet.

Die geo-globale BB ist nach den x/y-Achsen der verwendeten Karte ausgerichtet. Die 1. BB-Linie ist dabei parallel zur x-Achse und näher am y-Ursprung. Die weiteren Linien werden gegen den Uhrzeigersinn durchnummeriert.

5.1.3.2. Berechnung (Typ A)

Die Berechnung eines einzelnen Buffers vom Typ A betrifft immer Seite (vorne, hinten, links oder rechts) eines Objektes (Abbildung 2). Ziel ist es, eine Fläche zu schaffen, die jenseits oder im Anschluß der Objektbegrenzung liegt und die sowohl an der objektnahen als auch an der objektfernen Seite die Objektkonturen nachbildet.

Relevanzprüfung

Dazu muss zuerst herausgefunden werden, welche Punkte des Objektes relevant sind, das heißt, welche Punkte des Objektes auf der jeweiligen Seite liegen (Abbildung 3). Danach wird jeder dieser Punkte zweimal nach außen – also vom Objekt weg – projiziert, so dass einer der neuen Punkte auf der späteren objektfernen Bufferlinie und einer auf der späteren objektnahen Bufferlinie liegt (Abbildung 4).

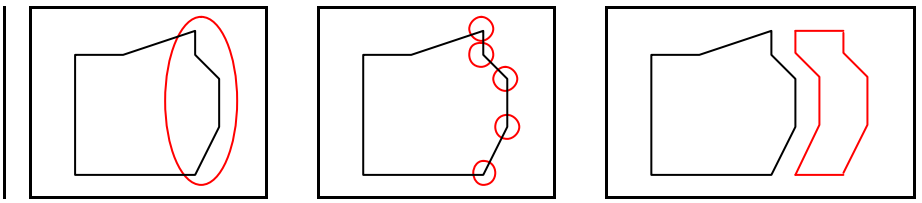

Abbildung 2 *Abbildung 3* *Abbildung 4*

Die erste Aufgabe des Berechnungsalgorithmus ist es also, herauszufinden, was es nun im speziell vorliegenden Fall heißt, wenn der Buffer auf der linken (rechten, vorderen, hinteren)

Seite des Objektes entstehen soll. Da es auch bei Objekten, die nicht mal annäherend rechteckig sind, Sinn macht, von den vier genannten Seiten zu sprechen, setzen wir BBs ein, um vier eindeutig definierte Seiten zu erhalten. Die Punkte, die „nahe“ (siehe unten) einer bestimmten BB-Seite sind, sind die auf der entsprechenden Seite liegenden Punkte.

Welche BB eingesetzt wird, ist abhängig von der Sichtweise (siehe → Sichtweisen statischer Raumausdrücke), die durch die Präposition und das Vorhandensein eines Betrachters abhängt. Generell entstehen z.B. die Buffer „neben“ auf den BB-Seiten 2 und 4, und je nachdem, ob es sich um eine intrinsische oder deiktische Sichtweise handelt, können die durch diese Angaben bezeichneten Objektpunkte sich stark unterscheiden (Abbildung 5 und 6).

Nachdem also die dem Fall entsprechende BB erzeugt wurde, wird für jeden einzelnen Objektpunkt überprüft, wie weit er von der relevanten BB-Seite entfernt ist. Nur wenn die Distanz kleiner oder gleich einer bestimmte, im entsprechenden Parametersatz der Präposition angegebene Entfernung, ist, wird der Punkt als relevanter Punkt bezeichnet. Die Entfernung ist als Prozentwert angegeben, und bezieht sich auf die senkrecht zur relevanten BB-Seite stehende BB-Seite, also sozusagen die „Tiefe“ des Objektes von der relevanten Seite aus gesehen.

Diese Vorgehensweise kann und soll bei konkaven Objektseiten durchaus dazu führen, dass extreme Einbuchtungen in der Bufferkontur nicht erscheinen, also geglättet werden (siehe Abbildung 7).

Anmerkung 1: Während der Implementierung erwies sich in praktischen Tests, dass die hier beschriebene Relevanzprüfung für deiktische Bufferberechnungen unschöne Ergebnisse ergab. Unschön heißt, dass die Auswahl der Punkte oftmals menschlich-intuitiv nicht nachvollziehbar war. Dies galt vor allem für Fälle, in denen das Referenzobjekt eine für eine menschlichen Betrachter annähernd rechteckig wirkende Grundform hatte und der Betrachter dann im spitzen Winkel zu den Seite des Referenzobjektes stand, also „auf eine Ecke blickte“.

Simples Herumexperimentieren mit den bereits implementierten Mitteln ergab, dass die Auswahl der relevanten Punkte deutlich näher an der menschlichen Wahrnehmung liegt, wenn man für die Relevanzprüfung bei deiktischer Sichtweise die Default-BB nimmt. Dabei muss allerdings beachtet werden, dass die relevante Seite der Default-BB neu berechnet werden muss, da die Durchnummerierung hier völlig abweichen kann (siehe Beschreibung der Default-BB). Dazu wird ähnlich wie der deiktischen BB berechnet, wo eine imaginäre Linie vom Betrachter zur Objektmitte die Default-BB schneidet. Die entsprechende Linie bekommt dann die Liniennummer 1 und alles weitere funktioniert wie üblich.

Anmerkung 2: Ein alternativer Ansatz zur Relevanzprüfung wäre eine Prüfung der Winkel auf der relevanten Objektseite. Auch in diesem Fall würde man Punkt für Punkt den Linienzug entlang gehen, und immer wenn die Aufnahme des aktuellen Punktes einen großen Winkel am vorigen Punkt bedeuten würde, lässt man den akuell untersuchten Punkt aus. Diese Alternative ist von uns nicht genauer untersucht worden, da die Idee dazu erst während der Projektpräsentation aufkam. {Vielleicht zusammen mit der Konturenproblematik unter Probleme erwähnen...}

Projektion

Jeder ausgewählte Objektpunkt wird dann zweimal nach außen, also weg von der Objektmitte, projiziert. Hierzu werden aus dem Präpositionsparametersatz zwei Distanz-

angaben in Prozent gelesen: Die Angabe für die objektnäheren und die objektferneren Punkte. Die Projektion erfolgt dann senkrecht zur relevanten BB-Linie. Für die Projektion wird in jedem Fall die sichtweisenabhängige BB benutzt, die relevante BB-Linie wird genauso bestimmt wie bei der Relevanzprüfung beschrieben.

Bufferbildung

Nachdem jeder relevante Punkt zweimal projiziert wurde, werden die Punkte zu einem durchgehenden Linienzug (Polygon) verbunden. Anschließend wird noch überprüft, ob der entstandene Buffer auf den beiden Tiefenseiten um einen im Präpositionsparametersatz festgelegt Winkel erweitert wird (siehe → Genauigkeit von Geltungsbereichen).

5.2. Die GUI

In der momentanen Form dürfte das Projekt fast ausschließlich über die GUI bedient werden, da die Kommandozeileneingabe und vor allem –ausgabe unübersichtlich ist und aufgrund der Natur der Daten (Koordinatenlisten) auch nicht übersichtlicher gestaltet werden kann.

Aufgerufen wird die GUI über das Modul `bgui.py`, wobei einer der beiden folgenden Parameter übergeben werden kann:

1. **-i <inifile>** Erlaubt es, eine alternative Datei mit Benutzereinstellungen einzulesen (an Stelle von `user.ini`)
2. **–d** Es werden keine Benutzereinstellungen gelesen

Weiteres zu den Einstellungsdateien siehe unter → INI-Dateien.

5.2.1. Anzeige

Das Hauptfenster der GUI unterteilt sich grob in drei Bereiche. Zuoberst befindet sich, wie üblich, eine Menüleiste über die verschiedene Funktionen der GUI angesteuert werden können. Darunter schließt sich der eigentlich wichtigste Teil der GUI an, der Kartenteil. Hier wird die momentan verwendete Karte, also die Objekte aus der momentanen Objektdatei, und eventuell noch ein Buffer und ein Betrachter angezeigt. Unter der Kartenanzeige befindet sich der Dialogbereich der GUI. Über die verschiedenen Funktionen in diesem Teil wird die GUI bedient.

Dieser untere Teil bedarf genauerer Beschreibung. Er besteht aus drei nebeneinander angeordneten Gruppen: Links befinden sich Eingabemöglichkeiten für die auszuwertende Präpositionalphrase und das Setzen des Betrachters. In der Mitte findet man Informationen zur Karte und zum aktuellen Objekt, und rechts erscheinen verschiedene Auswahlmöglichkeiten.

5.2.2. Bedienung

5.2.3. INI-Dateien

5.3. Parser

Beherrschte Phänomene
nicht beherrschte Phänomene

Weitere Module kurz vorgestellt

6. Ressourcen

In diesem Kapitel werden Aufbau und Inhalt aller Dateien beschrieben, die Bufferlo außer den Programmdateien benötigt. Dies sind vor allem die Präpositions-, Objekt- und Grammatikdatenbank. Als Datenbankformat haben wir XML gewählt, da es plattformunabhängig und einfach zu lesen und zu editieren ist. Da wir keine sehr großen Datenmengen zu erwarten hatten (die Präpositionen einer Sprache sind begrenzt), sollte auch das Parsen der XML-Dateien in akzeptabler Geschwindigkeit möglich sein. Theoretisch ist es jedoch kein Problem, das Format der Datenbanken zu ändern – lediglich die Lademodule müssten ausgetauscht werden (siehe → Programmstruktur).

Außer den verschiedenen Datenbanken wird in diesem Kapitel noch der Aufbau der INI-Dateien beschrieben. INI-Dateien werden einerseits verwendet, um die Einstellungen der GUI zu speichern; zum anderen gibt es eine INI-Datei, in der die Fehlerbeschreibungen der selbstdefinierten Fehler abgelegt sind.

6.1. Präpositionsdatenbank

In einer Präpositionsdatenbank-Datei können die Daten von Präpositionen und Modifikatoren gespeichert werden. Beim Berechnen eines Buffers kann immer auch angegeben werden, in welcher Präpositionsdatenbank die Daten der jeweiligen Präposition nachgeschlagen werden sollen. So könnte man also z.B. für unterschiedliche Sprachen unterschiedliche Präpositionsdatenbanken anlegen.

Im Folgenden wird jeder Tag der XML-Struktur mit eventuellem Inhalt und Attributen besprochen werden. Allgemeines Verständnis von XML wird dabei vorausgesetzt. Die Tags werden, sofern möglich, in der Reihenfolge besprochen, in der sie auftreten, wenn ein vollständiger Datensatz eingegeben wird. Wenn ein Tag an mehreren unterschiedlichen Stellen auftauchen kann, wird es jedoch nur einmal beschrieben. Selbstverständlich existiert auch eine DTD der Präpositionsdatenbank – sie hat den Namen `praeps.dtd`. [und befindet sich wo?] Während hier die semantische Bedeutung der Tags erklärt wird, dient sie zur Verdeutlichung der hierarchischen Struktur.

Die Tags im Einzelnen:

Name des Tags: `<spatials>...</spatials>`
Allgemeine Beschreibung: Diese Tags umschließen die gesamte Präpositionsdatenbank.
Inhalt: Die gesamte Präpositionsdatenbank
Attribute: keine

Name des Tags:
`<horizontal_projection>...</horizontal_projection>`
`<upright_projection>...</upright_projection>`

Allgemeine Beschreibung: Präpositionen und Modifikatoren können für die Berechnung von Flächen im Grundriss (horizontal projection) oder im Aufriss (upright projection) definiert werden. In der Datenbank werden zuerst alle Präpositionsdefinitionen für Grundriss, dann für

Aufriss angegeben. Das heißt, dass die folgenden Beschreibungen für beide Bereiche gelten.
Inhalt: Die Tags <preposition> und <modifikator>.
Attribute: keine

Name des Tags: <preposition>...</preposition>
Allgemeine Beschreibung: Diese Tags umschließen die Daten einer Präposition. Es können beliebig viele Präpositionsdefinitionen aufeinander folgen.
Inhalt: Die Tags <name>, <point_of_view>, <prep_grammar>, <prep_buffer> und <notes>.
Attribute: keine

Name des Tags: <name>...</name>		
Allgemeine Beschreibung: Der Name einer Präposition oder eines Modifikators. Besteht dieser aus mehreren Wörtern, so werden diese hier durch Leerzeichen getrennt hintereinander aufgeschrieben. Zur Kennzeichnung evtl. diskontinuierlicher Teile dient der <token>-Tag.		
Inhalt: s.o., zum Beispiel „links“ oder „neben“		
Attribute: 1		
Name:	**Beschreibung:**	**obligatorisch?**
id	Jede Präposition/ jeder Modifikator muss eine in der Präpositionsdatenbank einmalige ID haben. Es sind numerische und alphanumerische Zeichen erlaubt.	ja

Name des Tags: <point_of_view />		
Allgemeine Beschreibung: Für unterschiedliche Sichtweisen erzeugt eine Präposition unterschiedliche Flächen (siehe das erste Kapitel, Sichtweisen). Allerdings ist nicht jede Präposition für jede Sichtweise definiert. Mit Hilfe dieses Tags werden die Sichtweisen angegeben, für die die jeweilige Präposition Flächen erzeugt. Dazu wird es mit unterschiedlichen Attributswerten so oft wie nötig wiederholt. Es muss mindestens einmal angegeben werden.		
Inhalt: keiner		
Attribute: 1		
Name:	**Beschreibung:**	**obligatorisch?**
type	Gibt den Sichtweisentyp an. Erlaubte Werte sind intrinsic, deictic, geo-global und default. Intrinsic und deictic stehen für intrinsische und deiktische Sicht, geo-global wird für Präpositionen verwendet, die sich an einem geographisch genordeten Koordinatensystem orientieren (Himmelsrichtungen). Default ist für Präpositionen, deren Geltungsbereiche durch den Standpunkt eines Beobachters bzw. die Ausrichtung des Objekts nicht beeinflußt werden (z.B. an).	ja

Name des Tags: <prep_grammar>...</prep_grammar>
Allgemeine Beschreibung: Umschließt diverse Tags, die grammatische Informationen der Präposition beinhalten.
Inhalt: Die Tags <token>, <syntax>, <part_of_speech>, <case> und <combination>.
Attribute: keine

Name des Tags: `<token>...</token>`
Allgemeine Beschreibung: Wenn eine Präposition aus mehreren diskontinuierlichen Teilen besteht (z.B. an...entlang“), so können diese durch mehrere `<token>`-Einträge aufgelistet werden. Diese Informationen benötigt der Parser, um Präpositionalphrasen mit diesen Präpositionen richtig verarbeiten zu können. Wenn es sich nicht um eine diskontinuierliche Präposition handelt, gibt es nur einen `<token>`-Eintrag, in dem der Name der Präposition wiederholt wird.
Inhalt: Teil einer diskontinuierlichen Präposition; die gesamte Präposition, wenn sie nicht diskontinuierlich ist.
Attribute: keine

Name des Tags: `<syntax>...</syntax>`
Allgemeine Beschreibung: Mit Hilfe dieses Tags wird angegeben, wie die Syntax einer Präpositionalphrase (PP) mit dieser Präposition aussieht. Eine PP kann aus Nomen (`Noun`), Artikeln (`Art`), Präpositionen bzw. Präpositionsteilen (`Prep`) und Modifikatoren (`Mod`) zusammengesetzt sein. Diese Elemente werden nun in der Reihenfolge, in der sie in einer syntaktisch korrekten PP auftreten würden, aufgeschrieben. Weglassbare Elemente werden durch einen führenden * gekennzeichnet. Der Syntaxstring der Präposition „an...entlang“ sähe z.B. folgendermaßen aus: `*Mod Prep Art Noun Prep`. Dieser String beschreibt zum Beispiel die folgenden PP: „links an der Kirche entlang“. Bitte beachten: für jedes Vorkommen des Platzhalters `Prep` muss es einen Eintrag des `<token>`-Tags geben (siehe auch da).
Inhalt: ein String, bestehend aus den Platzhaltern `Prep`, `Art`, `Noun` und `Mod`. Ein * direkt vor einem Platzhalter kennzeichnet dieses Element als optional.
Attribute: keine

Name des Tags: `<part_of_speech>…</part_of_speech>`
Allgemeine Beschreibung: Hier kann die Wortart (part of speech) des Raumausdrucks angegeben werden. Dieser Eintrag wird vom Programm gelesen, im Augenblick aber nicht verwendet.
Inhalt: ein beliebiger String
Attribute: keine

Objektdatenbank
Grammatikdatenbank
INI-Dateien

Zusammenfassung

Erreichte Ziele
erwähnen: deiktisch/intrinsisch

Nicht erreichte Ziele
Weitere Präpositionstypen
Schraffur

- Modifikatoren

- Probleme
  - Projektion eigentlich unschön – konkave vs konvexe Objekte

Anhang

- Ideen zur Fortsetzung
  - dynamische Präpositionen
  - Tools
  - „Statistikfunktion“ in der GUI

- Systemvoraussetzungen
  - Pfade?
- Konventionen und Konstanten