Repräsentation von Sonardaten

Fortgeschrittenenpraktikum
am Lehrstuhl für Angewandte Mathematik und Mathematische Statistik
Priv.-Doz. Dr. S. Schäffler

Michael Gendreyzig — mike@leo.org

Frank Wolf — wolffr@in.tum.de

November 1998

Zusammenfassung

Diese Arbeit behandelt digitale Abbildungsmöglichkeiten von Umgebungen mobiler sonargestützter Roboter. Kernpunkte sind Visualisierung der Realwelt für den Betrachter und Aufbereitung der Sonardaten für eine algorithmische Wegplanung oder Positionsbestimmung. Grundlage der Berechnungen ist eine wahrscheinlichkeitstheoretisches Rechenmodell. Zusätzlich werden für Simulationen Konzepte zur Überprüfung von neu berechneten Raumpunkten vorgestellt (Guardians).

Eine Implementierungen in *C* dient als praktisches Beispiel der optischen Verknüpfung von Sonarsensor und Roboterfahrzeug. Die Modularität des Sourcecodes ermöglicht eine einfache Anbindung an Programme zur Lösung von Wegproblemen.

Erkärung:

Wir versichern, daß wir dieses Fortgeschrittenenpraktikum selbständig verfaßt und nur die angegebenen Quellen und Hilfsmittel verwendet haben.

München, den 29. Oktober 1998

Michael Gendreyzig Frank Wolf

Inhaltsverzeichnis

1 Einleitung

In der vorliegenden Arbeit wird das sonargestützte Kartographierungssystem autonomer mobiler Roboter in unbekannter und unstrukturierter Umgebung beschrieben. Das System benutzt Sonarentfernungsdaten, um eine mehrschichtige Beschreibung der Roboterumgebung zu modellieren. Sonarmessungen werden unter Zuhilfenahme von Wahrscheinlichkeitsverteilungen herangezogen, um leere und besetzte Gebiete zu bestimmen. Entfernungsmessungen von verschiedenen Standorten werden zu einer Sonarkarte zusammengefaßt. Die dabei verwendeten robusten Methoden vereinigen die Sensordaten so, daß sie auftretenden Unsicherheiten und Fehldaten gewachsen sind.

Die resultierenden zweisdimensionalen Karten können schließlich zur Pfadplanung und Navigation herangezogen werden.

Das Modell nach Elfes [Elf87] berechnet die Wahrscheinlichkeiten aller relevanten Raumpunkte, daß sie frei von Hindernissen sind (empty) und daß sie in irgendeiner Art besetzt sind (occupied). Nachfolgende Auflistung beschreibt die Veränderlichen und Konstanten, die für die Berechnung der Wahrscheinlichkeitswerte p_E und p_O jedes Punktes $P = (x, y, z)$ benötigt werden:

R Entfernungsmessung, die vom Sonarsensor zurückgegeben wird

ϵ maximaler Sonarmeßfehler

Ω Sensoröffnungswinkel

S $= (x_s, y_s, z_s)$ Standort des Sensors

δ Entfernung zwischen P und S

θ Winkel zwischen der Hauptachse des Sonars und P von S aus betrachtet

Wahrscheinlich freies Gebiet: das sind Punkte innerhalb des Sonarstrahls ($\delta < R - \epsilon, \theta \leq \frac{\Omega}{2}$)

$$p_E(x, y, z) = E_r(\delta) \cdot E_a(\theta)$$

$$E_r(\delta) = \begin{cases} 1 - (\frac{\delta - R_{min}}{R - \epsilon - R_{min}})^2 & f\ddot{u}r \quad \delta \in [R_{min}, R - \epsilon] \\ 0 & sonst \end{cases}$$

$$E_a(\theta) = 1 - (\frac{2\theta}{\Omega})^2 f\ddot{u}r \quad \theta \in [-\frac{\Omega}{2}, \frac{\Omega}{2}]$$

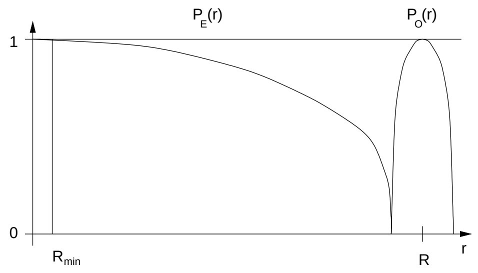
Besetzte Gebiete: Punkte am Ende des Sonarstrahls ($\delta \in [R - \epsilon, R + \epsilon], \theta \leq \frac{\Omega}{2}$)

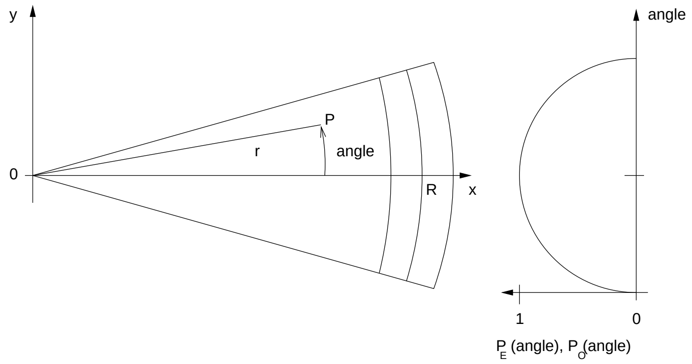
$$p_o(x, y, z) = O_r(\delta) \cdot O_a(\theta)$$

$$O_r(\delta) = \begin{cases} 1 - (\frac{\delta - R}{\epsilon})^2 & f\ddot{u}r \quad \delta \in [R - \epsilon, R + \epsilon] \\ 0 & sonst \end{cases}$$

$$O_a(\theta) = 1 - (\frac{2\theta}{\Omega})^2 f\ddot{u}r \quad \theta \in [-\frac{\Omega}{2}, \frac{\Omega}{2}]$$

Dies führt zu folgendender Verteilung, dargestellt in drei ebenen Schnitten:





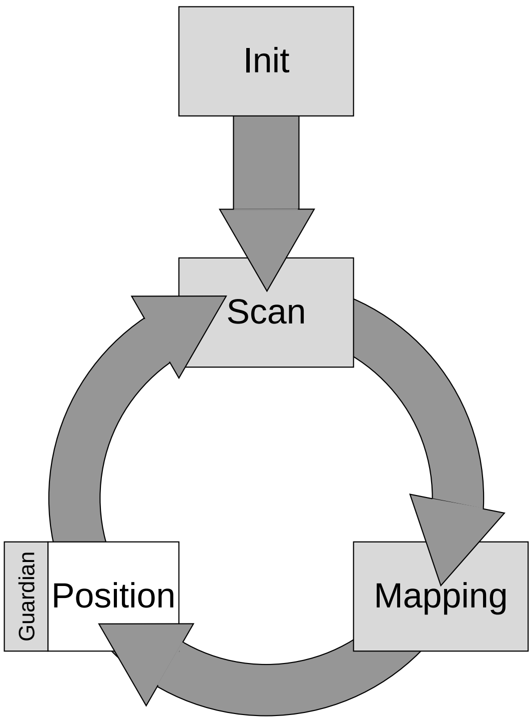
Zur Vereinigung von bisherigen Werten p_E und p_O mit den entsprechenden neu gemessenen Werten dienen die folgenden drei Zuweisungen (Superposition):

$$\begin{aligned}
p_E(Zelle) &:= p_E(Zelle) + p_E(Messung) - p_E(Zelle) \cdot p_E(Messung)\\
p_O(Messung) &:= p_O(Messung) \cdot (1 - p_E(Zelle))\\
p_O(Zelle) &:= p_O(Zelle) + p_O(Messung) - p_O(Zelle) \cdot p_O(Messung)
\end{aligned}$$

2 Analyse und Softwareentwurf

Die Erforschung eines unbekannten Raumes mittels eines mit einem Sonar ausgestatteten fahrbaren Roboters, stellt sich in unserem Ansatz wie folgt dar. Ein „Sonar-Scan“ des Raumes wird von der momentanen Position des Roboters durchgeführt. Die dadurch gewonnen Daten werden interpretiert und mit evtl. bereits gewonnenen Informationen zusammengefhrt und somit eine vorläufige „Karte“ des Raumes erzeugt. Nun wird mit Hilfe dieser „Karte“ des Raumes eine neue Position bestimmt, wohin sich der Roboter bewegen soll, um durch einen erneuten Scan von jener Stelle möglichst viele neue Informationen über den Raum zu gewinnen. Ein „Guardian“ überwacht den Weg des Roboters zu dieser neuen Stelle und informiert das System darüber, ob die Bewegung vollständig ausgeführt werden konnte oder ob der Roboter auf seinem Weg durch ein Hindernis gestört wurde. Dieser sich wiederholende Ablauf findet sein Ende, wenn der Raum hinlänglich bekannt ist.

Dieser Ansatz führt uns zu folgendem Programm-Schema:



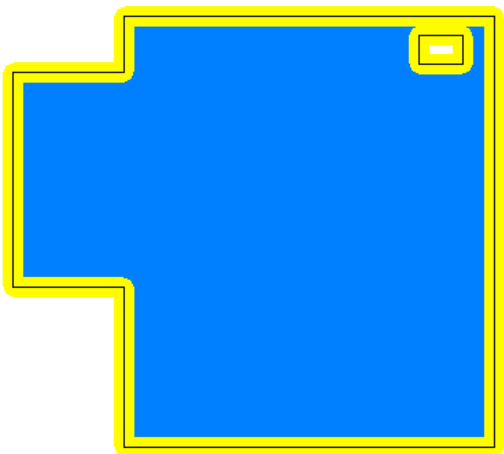
In der Initialisierungsphase werden die Programmparameter eingelesen, Variablen besetzt und Ausgabedateien angelegt. Mittels des Programmes „SoSimTab“ (SonarSimulation Tabelliert) [RT95] wird ein Sonar-Scan des Roboters simuliert. Die simulierten Werte werden in einer temporären Datei abgelegt. Die Daten werden nun in der Phase „Mapping“, gestützt auf den Modellansatz von Alberto Elfes [Elf87], interpretiert und mit bereits gewonnen Daten zusammengeführt. Die Bestimmung einer neuen Roboterposition geschieht momentan noch „von Hand“, in dem der Programmbenutzer anhand der letzen Visualisierung des Raumes einen neuen Punkt auswählt, vgl. hierzu Abschnitt 4.

Weitere Details sind dem nächsten Abschnitt zu entnehmen.

3 Implementierungen

Das Programm wird mit einer Datei (*.rsd) als Argument aufgerufen, die als zusätzliche Programmparameter neben weiteren Spezifizierungen des Roboter-Sensors, die Pfade zu den einzelnen, im Programmverlauf verwendeten, Dateien enthält.

In der folgenden Initialisierungsphase werden die globalen Variablen „sensor" und „rsd", deren Struktur in der Datei sbrwmap.h definiert ist, mit den Werten aus der angegebenen ViSoSim „Room"-Datei [RT95] belegt. Des weiteren werden zwei Bitmaps initialisiert. Die Erste wird im weiteren Verlauf des Programms die durch die Interpretation der Sensordaten gewonnene „Karte" des Raumes aufnehmen, die Zweite dient dazu, einen „Guardian", der die Bewegungen des Roboters auf Zusammenstöße mit Hindernissen überprüft, sehr vereinfacht zu simulieren. Dazu wird das Polygon das in der ViSoSim „Room"-Datei [RT95] den vorgegebenen Raum beschreibt in die Bitmap eingezeichnet und alle Bildpunkte, deren Abstand zu einer Mauer geringer als der Roboter-Radius (die Geometrie des Roboter wird vereinfacht als Kreis angenommen) ist, als „verboten" farblich gekennzeichnet. Daraufhin werden mit einem einfachen Floodfill-Algorithmus, ausgehend von der momentanen, ebenfalls in der „Room"-Datei definierten, Roboter Position alle theoretisch erreichbaren Punkte markiert.



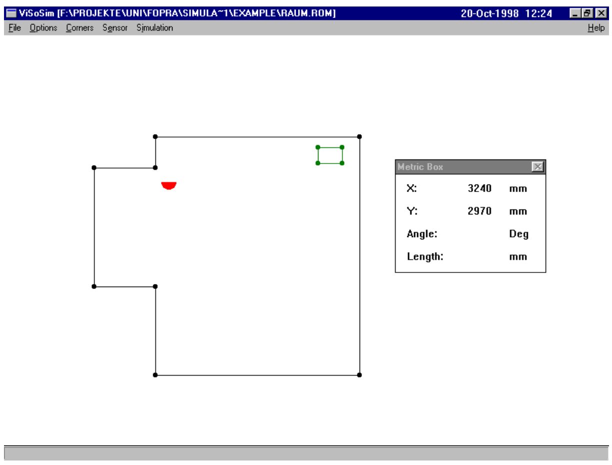
Im späteren Verlauf des Programms wird dann eine Bewegung des Roboters entlang der direkten Verbindung zwischen der alten und der gewünschten neuen Position vorausgesetzt und anhand der „Guardian"-Bitmap verfolgt, ob diese Strecke vollständig im „erlaubten" Bereich liegt. Ist dies nicht der Fall, so wird eine Warnung ausgegeben und die Position an welcher der Roboter mit dem Hindernis kollidiert an das Hauptprogramm übergeben.

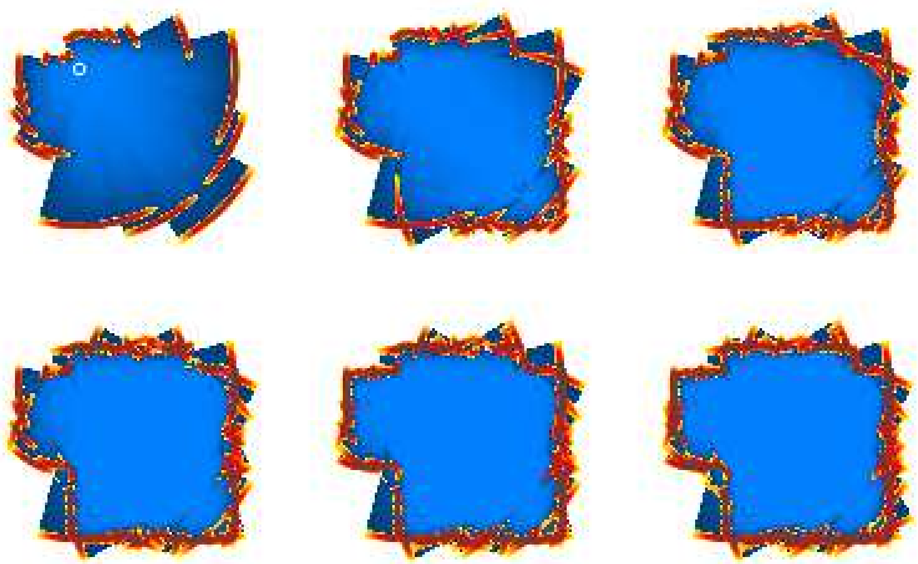
Der Scan von der aktuellen Position wird durchgeführt, indem das Programm SoSimTab [RT95] mit einer Kopie der „Room"-Datei gestartet wird, in welche die aktuelle Position des Roboters geschrieben wurde. Die simulierten Scan-Ergebnisse werden in ein Array eingelesen.

Danach werden für alle Bildpunkte der „Karten“-Bitmap mit Hilfe des Modells von Alberto Elfes [Elf87] die „occupied“- und „empty“-Wahrscheinlichkeiten errechnet und der bereits vorhandene Wert entsprechend angepaßt. Es werden keine getrennten Karten berechnet. Stattdessen wird lediglich die Differenz der beiden ermittelten Wahrscheinlichkeitswerte in der Karte gespeichert. Diese Vorgehensweise entspricht einem einfachen „Thrashholding“.

Nach der visuellen Analyse des berechneten Karte im BMP-Format wird über die Kommandozeile eine neue Position eingegeben, zu der sich der Roboter unter Beachtung des Guardians bewegt und einen neuen Scan durchführt.

Im Anschluß sind sechs Schritte einer Raumerkundung zu sehen, erzeugt mit dem Programm, das auf dem im Anhang befindlichen C-Quellcode basiert.





4 Ausblick

Die im letzten Abschnitt vorgestellte Implementierung läßt sich um weitere Funktionen erweitern. Eine automatische Berechnung neuer Positionen zur Raumvermessung und damit die verbundene Wegplanung sind wesentliche Komponeneten in Hinblick auf ein autonomes Erkundungssystem.

Die in den Karten gesammelten Informationen können in Algorithmen verwendet werden, die Positionen an Rändern von bereits bekannten Gebieten berechnen, von wo aus ein hoher Informationszuwachs durch neue Sonaruntersuchungen zu erwarten ist.

Auch der Weg zwischen aktueller und neu ermittelter Roboterposition kann durch einen Polygonzug verfeinert werden, dessen Berechnung sich auf vorangegangene Hindernismessungen stützt.

Literatur

[Elf87] Alberto Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3), June 1987.

[Pos91] Jeffrey A. Poskanzer. *ppm - portable pixmap file format.* ACME Laboratories, Berkeley, CA, September 1991.

[RT95] Michael Rempter and Martin Troll. Interactive simulation and visualization of sonar data. Technical Report 9508, Institute for Applied Mathematics and Statistics Technische Universität München, Arcisstr. 21, 80333 Munich, August 1995. from Copernicus Joint Research Project CP 93:5855, Information Processing for Active CIM Subsystems.