Lokalisierung in Sensornetzen

Seminararbeit für „Algorithmen in Sensornetzen“

Johannes Singler

Wintersemester 2004/2005

1 Einführung

1.1 Drahtlose Sensornetze

Ein Sensornetz besteht aus vielen, im Raum verteilten, so genannten Sensorknoten. Diese sind üblicherweise relativ klein, haben eine geringe Rechleistung und können nur mit begrenzter Reichweite drahtlos kommunizieren. In Abbildung 1.1 ist das mit einem kleinen Beispiel illustriert. Im Gegensatz zu klassischen IT-Systemen, die typischerweise eine *manuelle* Eingabe entgegennehmen, ermitteln diese Knoten über die eingebauten Sensoren die Beschaffenheit ihrer Umgebung selbst und verarbeiten sie weiter. So eignen sie sich z. B. zur Umweltüberwachung, aber auch zu militärischen Zwecken.

Die Ressourcen der Sensorknoten in Hinblick auf Rechenleistung und Kommunikationsmöglichkeiten sind eingeschränkt, einerseits durch die erwünschte Miniaturisierung und die erwünschten geringen Kosten pro Einheit, andererseits durch die knappe zur Verfügung stehende Energie. Wegen dieser besonderen Eigenschaften hat sich in den letzten Jahren ein eigenes Forschungsgebiet für Algorithmen entwickelt, die trotz dieser schwierigen Randbedingungen eine hohe Effizienz aufweisen. Es ist eine Forderung, dass komplexere Berechnungen verteilt ausgeführt werden, um die vorhandene Rechenleistung zu kumulieren und damit eine gute Skalierbarkeit zu erreichen. Es kann identische Hardware verwendet werden, was Kosten spart, außerdem wird die Energie im ganzen Netz gleichmäßig verbraucht, was z. B. solarbetriebenen Sensorknoten zu Gute kommt. Weiterhin ist eine direkte Kommunikation zwischen allen beteiligten Stationen oft nicht möglich, weil es Abschattungen gibt oder schlicht die Entfernung zwischen Sender und Empfänger zu groß ist. Im Normalfall kann ein Knoten daher nur mit wenigen Nachbarn Kontakt aufnehmen. So müssen Protokolle und Algorithmen zum Routing implementiert werden. Da sich die Sensoren u. U. auch bewegen können, oder auch komplett ausfallen, werden adaptive und fehlertolerante Verfahren eingesetzt.

Eine Familie dieser Verfahren ist das sogenannte geographische Routing für Netze, in denen die Knoten ihre relativen Positionen zueinander kennen. Dann können die Pakete zielgerichtet in eine geographische Richtung transportiert werden. Die Positionskenntnis kann auch für andere Aufgaben der Sensoren sehr nützlich sein. Da jedoch die Knoten typischerweise durch einen Zufallsprozess im Raum verteilt werden bzw. sich bewegen, sind die Positionen nicht a priori bekannt. Eine manuelle Eingabe ist zu aufwändig oder gar nicht möglich. Daher werden die Sensorknoten oft mit einem GPS-Empfänger ausgestattet, der es ihnen ermöglicht, ihre absolute Position jederzeit festzustellen. Für manche Fälle ist diese Ausstattung jedoch zu teuer und/oder zu energieaufwändig. In geschlossenen Räumlichkeiten ist diese Methode auch keine Lösung, denn es wird direkte

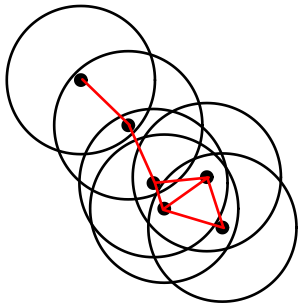


Abbildung 1.1: Kleines Sensornetz mit Funkradien und Linien zwischen Knoten mit Funkkontakt

Sicht auf mehrere Satelliten benötigt. Hier helfen dann höchstens lokale Funkbaken mit festgelegter Position, an denen sich die Sensoren orientieren können.

In dieser Ausarbeitung soll jedoch ein ganz anderer Ansatz vorgestellt werden. Die Sensoren besitzen nämlich immerhin die primitive geometrische Fähigkeit, den Abstand zu ihren wenigen nächsten Nachbarn im Funkradius zu messen. Dies kann über die Signalstärke oder -laufzeit geschehen. Nun können die Sensoren u. U. die relativen Positionen zueinander aus diesen Daten berechnen. Im Gegensatz zur Lokalisierung mittels GPS ist dies ein lokaler, verteilter Ansatz.

1.2 Problemdefinition

Die hier vorgestellten Algorithmen lösen das *Sensor-Layout-Problem*:

Gegeben seien eine Menge von Sensorknoten mit unbekannter Position und eine Methode, mit der jeder Sensorknoten seinen Abstand zu einigen nahen Knoten schätzen kann. Bestimme die relativen Koordinaten jedes Sensors mit Hilfe von ausschließlich lokaler Sensor-zu-Sensor-Kommunikation. Die Gesamtheit dieser Koordinaten nennt sich das Layout *des Sensornetzes.*

Weil zunächst einmal keinerlei Wissen über die absolute Position im Raum verfügbar ist, können nur Positionen relativ zueinander festgestellt werden. Daher ist das Ergebnis im Zweidimensionalen höchstens bestimmbar bis auf eine beliebige Translation, eine Rotation und eine Spiegelung. Die Skalierung ist jedoch durch die gemessenen Abstände fix. Um auch die absoluten Positionen errechnen zu können, braucht man jetzt nur noch die absoluten Koordinaten von drei der Sensorknoten zu kennen.

Leider sind aber auch die relativen Koordinaten im Allgemeinen nicht eindeutig. Aussagen über die sogenannte infinitesimale Starrheit (Knoten lassen sich nicht stetig verschieben) bzw. globale Starrheit (die Lösung ist global eindeutig) sind sehr schwierig zu treffen. Selbst wenn sämtliche Abstände algebraisch unabhängig sind (eine starke Forderung), muss der Sichtbarkeitsgraph mindestens 6-fach zusammenhängend sein, um mit Sicherheit global starr zu sein. Für eine genauere Untersuchung dieses Sachverhalts

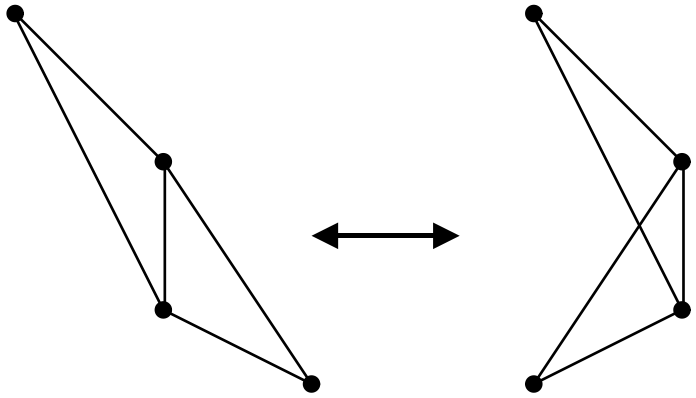


Abbildung 1.2: Einfaches Beispiel für einen Umschlag, die richtige Lösung ist nicht eindeutig.

siehe [3]. Ein weiteres Ergebnis ist, dass es sogar für eindeutig lösbare Fälle NP-hart ist, diese eine Lösung zu finden [7]. Die größte Gefahr für die Rekonstruktion des korrekten Layouts sind „Umschläge“, bei denen ein Teil des Netzes entlang einer Linie über den Rest „herüberklappt“ (siehe Abbildung 1.2). Es hat sich als Hauptproblem herausgestellt, diese Umschläge zu vermeiden. Wären sämtliche paarweisen Abstände zwischen den Sensorknoten gegeben, wäre das Problem übrigens ziemlich einfach. Es lässt sich ein intuitiver Algorithmus angeben, das Layout zu (re)konstruieren, siehe [1].

Weiterhin sind von der Längenmessung Ungenauigkeiten zu erwarten. Auch diese dürfen das Ergebnis nicht gleich entwerten. Es muss also letztlich die Abweichung von der Realität minimiert werden. Auch für die Aussage, ob zwei Stationen miteinander kommunizieren können, ist in Abhängigkeit vom Abstand nur eine probabilistische Aussage möglich. Physikalische Unwägbarkeiten können den Empfang zufällig stören.

1.3 Gliederung

Ich werde im folgenden Kapitel zunächst einen allgemeinen Überblick über Lösungsansätze liefern und dann genauer auf den Algorithmus von Gotsman und Koren [4] eingehen, was den Hauptteil dieser Seminararbeit ausmacht. Dieser Algorithmus basiert auf Ergebnissen aus dem Graphenzeichnen, so genannten Spektral-Layouts. Die experimentelle Resultate damit werden dann mit mit denen eines früheren Algorithmus von Priyantha et al. [8], genannt Anchor Free Layout (AFL), verglichen.

2 Lösungsansätze

Frühe algorithmische Ansätze benötigen für eine gewisse Teilmenge der Sensoren schon gegebene Positionen, um die restlichen berechnen zu können. Man unterscheidet daher folgende zwei Alternativen:

anker-basiert Die absoluten Positionen eines gewissen Anteils (z. B. 10%) von Sensoren muss vorher bekannt sein. Da diese dann anderweitig bestimmt werden müssen, wird das Problem nicht vollständig gelöst, sondern nur vereinfacht.

anker-frei Die relativen Positionen werden rein aus den bekannten Abständen berechnet. Nur, falls man absolute Koordinaten möchte, muss man noch die Positionen von wenigen Sensoren (drei Stück im Zweidimensionalen) kennen.

Natürlich sind anker-freie Algorithmen den anker-basierten vorzuziehen. Experimente haben jedoch sowieso gezeigt, dass verankerte Sensoren nicht wirklich helfen. Die Ergebnisse waren trotzdem zum Großteil unbrauchbar, wenn man nicht schon sowieso fast alle Positionen eingeben musste.

Der nächste grundlegende Unterschied ist die Reihenfolge der Positions-Festlegung, siehe auch Abbildung 2.1:

inkrementell Die Positionen werden Knoten für Knoten nacheinander einmalig festgelegt und dann nicht mehr verändert.

simultan Die Positionen aller Knoten werden gleichzeitig verändert, bis eine große Übereinstimmung mit der Zielvorgabe erreicht ist.

Die simultane Variante ist besser, da die inkrementelle zu anfällig ist für lokale Minima. Ein einmal gemachter Fehler bei der Positionierung kann nie wieder korrigiert werden.

Eine wichtige nichtfunktionale Eigenschaft ist, ob der Algorithmus verteilt arbeitet.

zentral Alle Eingabedaten werden zu einem Knoten transportiert, dort verarbeitet und die Ergebnisse wieder über das Netz verteilt.

verteilt Alle Knoten berechnen parallel ihre eigene Position. Datenaustausch findet nur mit den direkten Nachbarn statt, allerdings ergibt sich indirekt eine Informationsausbreitung über das ganze Netz.

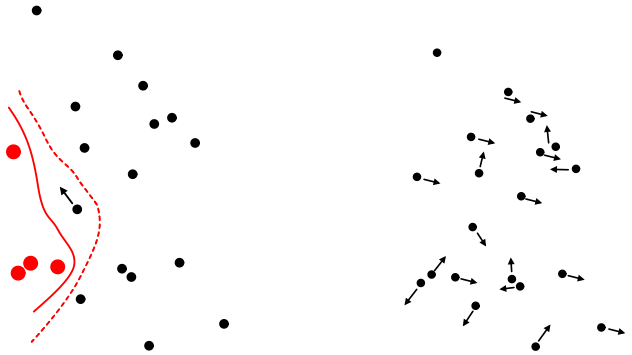


Abbildung 2.1: Inkrementelle und simultane Lokalisierung

Ein verteilter Algorithmus ist vorzuziehen, denn dies macht den Algorithmus gut skalierbar. Zudem soll zur Berechnung Kommunikation nur mit den direkt erreichbaren Nachbarn verwendet werden, damit man nicht schon an dieser Stelle ein Routing-Verfahren braucht.

3 Der Algorithmus von Gotsman und Koren

3.1 Idee

Die offensichtliche Modellierung des Problems ist ein kantengewichteter Graph. Jeder Knoten im Graphen entspricht dabei einem Sensorknoten. Zwei Knoten sind genau dann durch eine Kante verbunden, wenn sie über Funk miteinander kommunizieren können. Das Kantengewicht entspricht dabei der gemessenen Länge. Es wird im Folgenden immer davon ausgegangen, dass der Graph ungerichtet und damit die Relation symmetrisch ist. Dies lässt sich im Zweifelsfall dadurch erreichen, dass ein Knoten eine Kante nur zu den Nachbarn einführt, zu denen bidirektionale Kommunikation möglich ist, die also auf eine Anfrage antworten. Dies ist eine konservative Abschätzung.

Das Problem besteht nun darin, die Knoten so im Raum einzubetten, dass die gewünschten Abstände möglichst genau eingehalten werden. Zudem sollen aber Knoten, die nicht mit einer Kante verbunden sind, einen Abstand größer der Kommunikationsreichweite haben. Im Allgemeinen sollten die Algorithmen in beliebigen Dimensionen funktionieren. Hier werden wir uns aber meistens auf den zweidimensionalen Fall beschränken, da die Beispiele dann schön anschaulich sind. Dies ist auch in der Praxis ein oft benötigter Fall.

Formal ergibt sich folgendes: Ein Graph $G(\{1, ..., n\}, E)$ mit Kantenmenge E und für jede Kante $\langle i, j\rangle$ die gemessene Länge ℓ_{ij}. Die herauszufindenden Positionen seien mit p_i notiert. Das Problem lässt sich nun formulieren zu: Finde eine Belegung für die p_i im Raum, so dass gilt:

$$\|p_i - p_j\| = \ell_{ij} \qquad \text{falls} \qquad \langle i, j\rangle \in E \tag{3.1}$$

$$\|p_i - p_j\| \geq r \qquad \text{falls} \qquad \langle i, j\rangle \notin E \tag{3.2}$$

wobei $r = \max_{i,j}(\ell_{ij})$, also der maximal vorkommende gemessene Abstand. Sinnvoll ist das alles jedoch nur, wenn der Graph zusammenhängend ist. Ansonsten kann man ihn einfach in Zusammenhangskomponenten aufteilen und getrennt bearbeiten. Es existiert dann sowieso keinerlei Information über die Lage von Knoten in verschiedenen dieser Komponenten zueinander.

Letztlich hat man also ein Problem, das aus einem anderen Gebiet, dem Graphenzeichnen, schon bekannt ist. Die Autoren Gotsman und Koren stammen auch aus dieser Forschungsrichtung und haben erstmals Techniken aus diesem Gebiet bei der Lokalisierung in Sensornetzen angewendet. Im Gegensatz zum Graphenzeichnen, bei dem es meist um eine möglichst ästhetische Einbettung geht, möchte man hier die etwas über die

Realität herausfinden. Damit gibt es im Wesentlichen nur eine richtige Lösung, das Problem hier stellt also einen höheren Anspruch. Andererseits weiß man immerhin, dass eine Lösung existieren muss, zumindest innerhalb der Messfehler. Es kommen jedoch noch Forderungen aus dem Gebiet der Sensornetze hinzu, nämlich z. B. der Wunsch nach verteilter Berechnung. Dieses Kriterium haben Algorithmen aus dem Graphenzeichnen bisher nicht berücksichtigt, hier musste also noch Forschungsarbeit geleistet werden.

3.2 Überblick über den Algorithmus

Der Algorithmus von Gotsman und Koren hat alle der im vorigen Kapitel genannten wünschenswerten Eigenschaften. Er ist

- *anker-frei*
- *simultan* und
- *verteilt.*

Insgesamt könnte man ihn als „verteilten Graphen-Zeichnen-Algorithmus“ subsummieren.

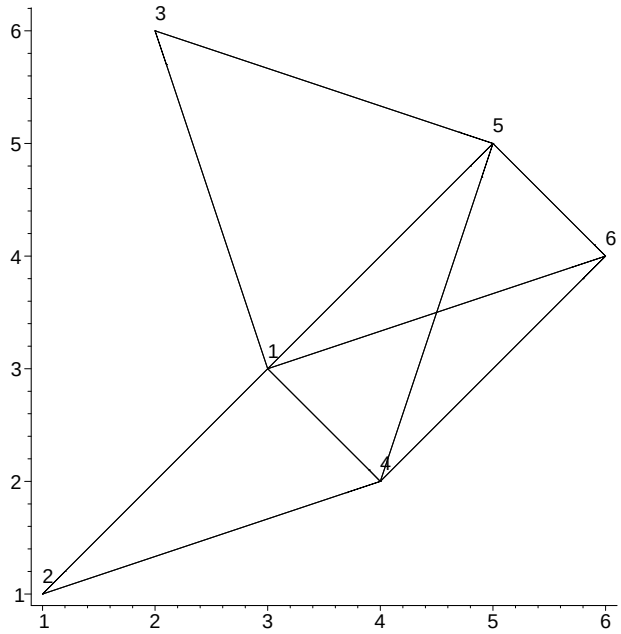
Auf Grund von Messungenauigkeiten ist es nicht sinnvoll, mit kombinatorischen Verfahren nach der „perfekten Lösung“ zu suchen. Stattdessen wird mit einer Optimierungsstrategie die Lösung immer besser angenähert. Eine naheliegende Bewertung der Lösung ist die sogenannte „Localized Stress Energy“, welche durch die perfekte Lösung minimiert wird ($= 0$):

$$Stress(p) = \sum_{\langle i,j\rangle \in E} (d_{ij} - \ell_{ij})^2 \tag{3.3}$$

Dabei bezeichnen die d_{ij} die Abstände, die durch die Lösung bestimmt werden. Leider ist es mathematisch schwierig, für diese Gleichung eine gute Lösung ohne Umschläge zu finden. Betrachten wir daher die Alternative

$$E(p) = \frac{\sum_{\langle i,j\rangle \in E} w_{ij} \|p_i - p_j\|^2}{\sum_{i<j} \|p_i - p_j\|^2} \tag{3.4}$$

worin p der Vektor der errechneten Positionen ist. Durch die Koeffizienten w_{ij} gehen dabei die gemessenen Abstände „reziprok“ ein. Je größer der gewünschte Abstand, desto kleiner muss w_{ij} sein. Dann geht der entsprechende Abstand geringer in die Summe ein und sie bleibt auch für einen größere Abstand relativ klein. Die Normierung im Nenner verhindert, dass alle Knoten auf einen Punkt fallen, was keine sinnvolle Lösung ist. Als Formel für die w_{ij} eignen sich z. B. $w_{ij} = exp(-\alpha \ell_{ij})$ oder $w_{ij} = \frac{1}{1+\ell_{ij}}$ für ein festes $\alpha > 0$. Wie sich herausstellen wird, gibt es effizientes Verfahren, dass diese Bewertungs-Funktion minimiert. Ein Beispiel-Netz mit zugehöriger Matrix W zeigt Abbildung 3.1.



$$W = \begin{pmatrix} 0.44 & -0.06 & -0.04 & -0.24 & -0.06 & -0.04 \\ -0.06 & 0.1 & 0 & -0.04 & 0 & 0 \\ -0.04 & 0 & 0.08 & 0 & -0.04 & 0 \\ -0.24 & -0.04 & 0 & 0.38 & -0.04 & -0.06 \\ -0.06 & 0 & -0.04 & -0.04 & 0.38 & -0.24 \\ -0.04 & 0 & 0 & -0.06 & -0.24 & 0.34 \end{pmatrix}$$

Abbildung 3.1: Beispiel-Netz mit zugehöriger Gewichtsmatrix

Der Algorithmus von Gotsman und Koren geht nun in zwei Stufen vor:

1. **Initiales Layout (Spektral-Layout)** Durch Minimierung von $E(p)$ wird ein möglichst umschlagsfreies initiales Layout erzeugt, dass unabhängig vom Anfangszustand gegen eine bestimmte Lösung konvergiert. Jedoch gibt es die gemessenen Abstände nicht gut wieder. Basis des Verfahren ist, dass die minimale Lösung von $E(p)$ eine Kombination aus Eigenvektoren zu einer bestimmten Matrix ist. Daher nennt man dieses Layout auch Spektral-Layout.

2. **Energie-Optimierung** Ausgehend vom initialen Layout wird nun die zweite Zielfunktion $Stress(p)$ minimiert, um realistische Abstände zu erhalten. Der Erfolg basiert stark auf einem guten Ergebnis des ersten Schritts, erzeugt dann aber oft sehr gut angenäherte Lösungen. Als Verfahren kommt entweder der klassische Gradientenabstieg oder ein verbessertes Verfahren namens „Majorization“ zum Einsatz.

3.3 Initiales Layout

Die Spektral-Methode zum Zeichnen von Graphen wurde von Hall [2] eingeführt. Zwei Eigenschaften machen diese Methode besonders attraktiv. Zum einen ergibt sie eine mathematisch exakte Lösung des Layout-Problems, während fast alle andere Verfahren NP-hart sind und damit nur approximiert werden können. Zum anderen lässt sich diese Lösung effizient und schnell berechnen.

Für die weitere Beschreibung benötigen wir einige Definitionen:

- Die Diagonalmatrix D hat den verallgemeinerten Grad eines jeden Knoten $deg_i := D_{ii} := \sum_{j:\langle i,j\rangle \in E} w_{ij}$ als Diagonaleinträge.
- Die Laplacesche Matrix $L = D - W$ mit $W = (w_{ij})_{i,j}$ hat somit Reihen- und Spaltensumme 0.

Ziel ist es nun, $E(p)$ zu minimieren. Dazu setzen wir zunächst einmal alle Koordinaten bis auf die der ersten Dimension gleich 0. Der Vektor der ersten Koordinaten über alle Knoten heiße jetzt x. Da $E(x) \geq 0$ gilt, ist $E(x)$ für $x_1 = \ldots = x_n = c$ mit einer beliebigen Konstante c minimal. Das würde bedeuten, dass alle Knoten auf einen Punkt fallen, was keine sinnvolle Lösung des Problems darstellt. Daher fordern wir eine gewisse Varianz. Wir wählen geschickt $Var(x) = \frac{1}{n}\sum_i (x_i - \overline{x})^2 \stackrel{!}{=} \frac{1}{n}$.
Da außerdem die Lösung invariant bzgl. Translation ist, legen wir den Schwerpunkt aller Knoten willkürlich auf 0 fest, um einen Freiheitsgrad zu eliminieren: $\overline{x} = 0$

Dann ergibt sich aus beidem:

$$x^T x \stackrel{!}{=} 1 \qquad\qquad x^T \cdot 1_n \stackrel{!}{=} 0 \tag{3.5}$$

Diese Bedingungen und die Zielfuntion optimieren nun Folgendes: Knoten i und j, die durch eine Kante $\langle i, j\rangle$ verbunden sind, sollen umso näher beisammen sein, je größer das Kantengewicht w_{ij} ist. Dabei soll jedoch global gesehen eine gewisse Spreizung bestehen bleiben und die Punkte außerdem gleichmäßig um den Ursprung verteilt sein.

Satz Das globale Minimum von $E(x)$ unter Bedingungen (3.5) ist der Eigenvektor zum

- kleinsten positiven Eigenwert von L
- größten Eigenwert außer 1 von $D^{-1}W$.

Beweis für die erste Aussage:

L ist symmetrisch und positiv semi-definit. Daraus folgt, dass alle Eigenwerte von L reell und nichtnegativ sind. Außerdem existiert damit eine Orthogonalbasis $\{e_1, \ldots, e_n\}$ aus Eigenvektoren. Der Eigenwert $\lambda_1 := 0$ ist Eigenwert zum Eigenvektor 1_n. Seien die anderen Eigenwerte aufsteigend nummeriert $0 < \lambda_2 \leq \lambda_3 \leq \ldots \leq \lambda_n$.

Zu minimieren ist dann:

$$E(x) = \frac{x^T L x}{x^T x} = x^T L x = \sum_{i<j} w_{ij} \|x_i - x_j\|^2$$

Behauptung: $E(e_2) = \lambda_2$ ist minimal.

Beweis: Stelle x, beliebig orthogonal zu 1_n in der Eigenbasis dar, der Anteil von $e_1 = 1_n$ spielt dann keine Rolle, $x = \sum_{i=2}^{n} \alpha_i e_i$. Es muss nun gelten

$$1 = x^T x = \left(\sum_{i=2}^{n} \alpha_i e_i\right)^T \left(\sum_{i=2}^{n} \alpha_i e_i\right) = \sum_{i=2}^{n}\sum_{j=2}^{n} \alpha_i \alpha_j \underbrace{e_i e_j}_{\delta_{ij}} = \sum_{i=2}^{n} \alpha_i^2$$

$$x^T L x = \left(\sum_{i=2}^{n} \alpha_i e_i\right)^T \left(\sum_{i=2}^{n} \alpha_i L e_i\right) = \sum_{i=2}^{n}\sum_{j=2}^{n} \alpha_i \alpha_j \lambda_j \underbrace{e_i e_i}_{\delta_{ij}} =$$

$$\sum_{i=2}^{n} \alpha_i^2 \lambda_i \overset{\lambda_2 \leq \lambda_i}{\geq} \underbrace{\sum_{i=2}^{n} \alpha_i^2}_{=1} \lambda_2 = \lambda_2$$

$$\Rightarrow \qquad x^T L x \geq \lambda_2 \wedge e_2^T L e_2 = \lambda_2 \quad \Rightarrow \quad E(e_2) = \lambda_2 \quad \text{minimal.} \qquad \square$$

Eine Verbesserung in der Kovergenzgeschwindigket ergibt sich mit der so genannten grad-normalisierten Variante:

Zu minimieren ist dann $x^T(D^{-1}W)x$ unter den Bedingungen $x^T D x = 1$ und $x^T D 1_n = 0$.

Hier spielt dann die D-Orthogonalität eine Rolle. In diesem Fall muss dann die zweite Aussage des obigen Satzes verwendet werden. Dessen Herleitung findet sich genauer und ausführlicher in [5].

Zur konkreten Berechnung der Eigenvektoren wird eine Potenz-Iteration (power iteration) über $M = I + D^{-1}W$ verwendet, also $Mx, M^2x, M^3x, \ldots$. Diese Folge konvergiert gegen den größten Eigenvektor. Die lokale Berechnungsvorschrift für jeden Knoten ergibt sich damit zu

$$p_i \leftarrow \frac{1}{2}\left(p_i + \frac{\sum_{\langle i,j\rangle \in E} w_{ij} p_i}{deg_i}\right) \tag{3.6}$$

Bildlich gesprochen ist das die Mittelung der bisherigen Position mit dem Schwerpunkt der Nachbarn. Der entscheidende Vorteil dieser Formel ist, dass sie nur Information der Nachbarn in direkter Kommunikationsreichweite benötigt. Damit lässt sich die Iteration verteilt durchführen. Sie konvergiert nun also gegen das gewünschte Minimum. Der sich ergebende Vektor bestimmt die Positionskoordinaten in der ersten Dimension x. Wie sich herausstellt, kann man die Eigenvektoren zu den weiteren Eigenwerten für die Koordinaten in den höheren Dimensionen nehmen, also z. B. den Eigenvektor zum zweitgrößten Eigenwert außer 1 von $D^{-1}W$ als y. Die Berechnung kann mit der gleichen Vorschrift wie oben durchgeführt werden. Es muss nur zu jedem Zeitpunkt der Vektor D-orthogonal zu sämtlichen vorher berechneten Eigenvektoren x_k gehalten werden, also $x^T x_k = 0$. Der initiale Vektor muss diese Bedingung auch schon erfüllen.

Eine ausführliche Behandlung dieser Verfahren aus der numerischen Linearen Algebra bietet [6].

Jetzt stellt sich nur noch die Frage, warum ausgerechnet diese Spektral-Methode für die erste Stufe des Algorithmus verwendet wird. Erfahrungsgemäß ergeben sich daraus meistens umschlagsfreie Layouts, was für das Herausfinden der realitätsgetreuen Lösung äußerst wichtig ist.

Das Ergebnis der ersten Stufe für das Beispiel-Netz ist in Abbildung 3.2 abgebildet.

3.4 Energie-Minimierung

In dieser zweiten Stufe erfolgt eine Rückbesinnung auf die erste Bewertungsfunktion „Localized Stress Energy“. Hierbei verwendet man den Gradientenabstieg oder eine verwandte Methode, „Majorization“. Beide sind eigentlich anfällig für lokale Minima, aber man hofft, durch das initiale Layout bereits im „richtigen Tal“ zu sein.

Für den Gradientenabstieg ergibt sich als Iterationsformel

$$p_i(t+1) = p_i(t) + \delta \sum_{j:\langle i,j\rangle \in E} \frac{p_j(t) - p_i(t)}{d_{ij}(t)}(d_{ij}(t) - \ell_{ij}) \tag{3.7}$$

wobei p noch einmal implizit in den d_{ij} versteckt ist. Für „Majorization“ sieht die Formel etwas anders aus:

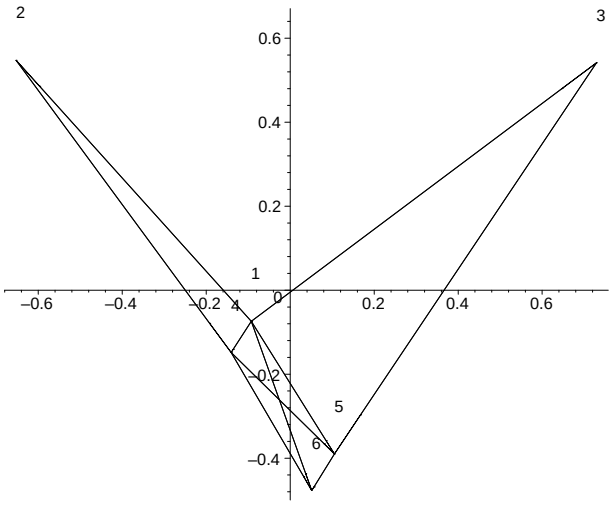


Abbildung 3.2: Initiales Layout für das Beispiel-Netz

$$p_i(t+1) = \frac{1}{deg_i} \sum_{j:\langle i,j\rangle \in E} (p_j(t) - \ell_{ij}(p_i(t) - p_j(t))\text{inv}(d_{ij}(t)) \tag{3.8}$$

Dabei ist $\text{inv}(x) := \frac{1}{x}$ bis auf $\text{inv}(0) := 0$. Setzt man jedoch $\delta := \frac{1}{deg_i}$ und $\text{inv}(x) = \frac{1}{x}$, dann sind die Formeln fast identisch.

Wiederum sind beide verteilt berechenbar und benötigen nur Kenntnis über die Werte adjazenter Knoten.

Das Ergebnis für das Beispiel-Netz zeigt Abbildung 3.3. Bis auf eine Drehung stimmt es sehr genau mit dem Ausgangsbild überein.

3.5 Komplexität und Geschwindigkeit

Zunächst einmal stellt sich die Frage nach der Terminierung des Algorithmus. Bei Algorithmen, die gegen eine gewisse Lösung konvergieren, bricht man normalerweise ab, wenn über eine gewisse Anzahl Schritte keine oder nur eine zu kleine Verbesserung des Ergebnisses erreicht worden ist. Dies ist hier aber nicht so einfach möglich, da man dazu globales Wissen benötigen würde. Die Autoren haben die einfache Variante gewählt, die Anzahl der Iterationen von vornerherein festzulegen. Diese Zahl steigt proportional zur Anzahl Knoten und liegt größenordnungmäßig bei $10n$.

Durch die begrenzte Funkreichweite gibt es bei einer als konstant angenommenen Dichte der Sensorknoten nur $O(1)$ viele Kanten pro Knoten, also insgesamt $O(n)$ Kanten. Damit ergibt sich also insgesamt eine Arbeit von $O(n^2)$, aber nur eine Laufzeit von $O(n)$ auf Grund der parallelen Berechnung.

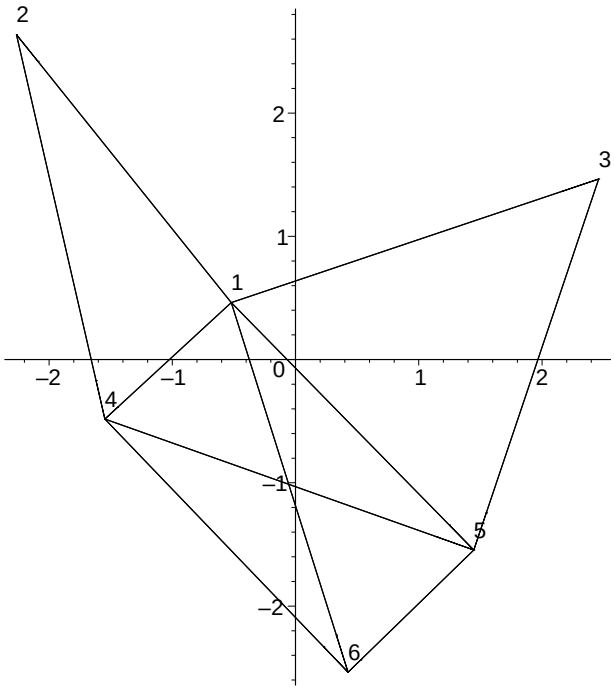


Abbildung 3.3: Layout fürs Beispiel-Netz nach der Energie-Minimierung

Man kann auch auf eine Terminierung des Algorithmus komplett verzichten und ihn neben der eigentlichen Aufgabe des Sensornetzes „ewig“ nebenher laufen lassen. Dann reagiert dieser automatisch auf Positionsänderungen und führt die Schätzung nach.

3.6 Verfeinerungen

Die Autoren von [4] schlagen zu diesem gut strukturierten Ansatz noch ein paar Verfeinerungen vor, die sich in der Praxis bewährt haben.

Zunächst ist es sehr wichtig, numerische Instabilitäten zu vermeiden. Daher sollten die Vektoren in der Eigenwertberechnung nach jedem Schritt neu zu den zuvor berechneten (D-)orthogonalisiert und außerdem normiert werden, weil sie sonst durch die exponentielle Verlängerung oder Verkürzung über die Iterationen den Rahmen von Fließkomma-Formaten sprengen könnten. Dies wäre rein theoretisch nicht notwendig.

Weiterhin ist es auch möglich, die zuvor in klarer Reihenfolge dargestellten Stufen 1 (initiales Spektral-Layout) und 2 (Energie-Minimierung) ineinander zu verzahnen, was zunächst nicht offensichtlich ist. Nach einem Interationsschritt von Stufe 1 folgen dann z. B. 30 Schritte der Stufe 2, um näher an das reale Ergebnis heranzukommen.

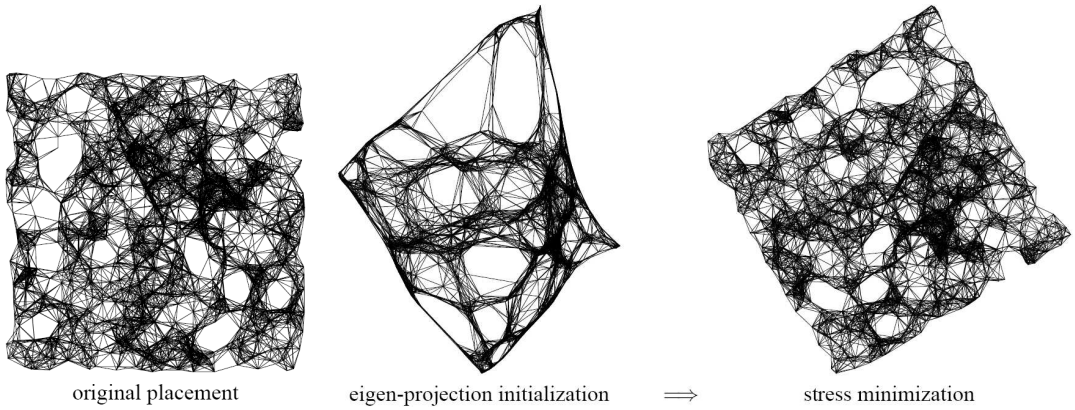


Abbildung 3.4: Beispiel aus [4], das Ergebnis stimmt bis auf Rotation sehr gut mit dem Ausgangsbild überein.

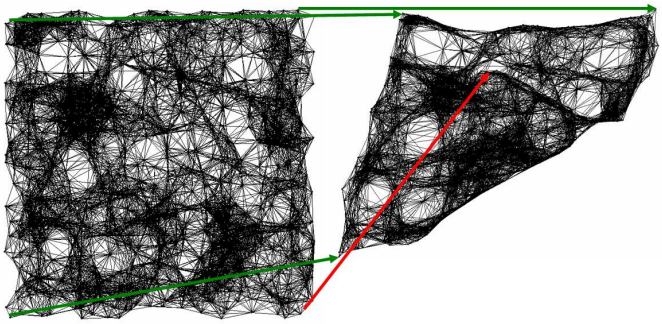


Abbildung 3.5: Beispiel für einen Umschlags-Fehler, erstellt mit dem Programm der Autoren von [4].

4 Experimentelle Ergebnisse und Bewertung

4.1 Vergleichspartner

Die Autoren von [4] vergleichen ihren Algorithmus mit dem besten bis dahin bekannten Vorgänger, dem so genannten *Anchor Free Layout (AFL)* aus [8]. Dies war der erste Algorithmus, der simultan und anker-frei arbeitete. Auch er ist zweistufig, die erste Stufe wird allerdings nicht verteilt berechnet. In der zweiten Stufe gleicht er mit einem Gradientenabstieg eigentlich dem Algorithmus von Gotsman und Koren, wobei jener durch die Verwendung der Majorization-Methode eine graduelle Verbesserung enthält.

Das initiale Layout wird folgendermaßen berechnet:

- Bestimme über graphentheoretische Distanz (z. B. Länge des minimalen Wegs) einen „zentralen“ Knoten. Finde weitere vier Knoten, die von diesem Zentrum aus möglichst weit in (den) vier Himmelsrichtungen liegen, die Verbindunglinien sich also möglichst rechtwinklig kreuzen. Da die Positionen der Knoten zu diesem Zeitpunkt natürlich noch völlig unbekannt sind, muss man das auch rein aus der Struktur des Graphen approximieren.
- Schätze für jeden Knoten durch seine graphentheoretischen Distanzen zu den fünf Referenzknoten den Abstand zum Zentrum und den Winkel des Strahls in seine Richtung. Damit erhält man im initialen Layout für jeden Graphen Polarkoordinaten. Ein Beispiel-Ergebnis zeigt Abbildung 4.1.

4.2 Vergleichsmaß

Als Maß für die Güte eines Algorithmus bietet sich zunächst einmal die Summe der Differenzen der gefundenen Abstände zu denen der gemessenen Abstände an; je kleiner, desto besser. Diese hat jedoch den Nachteil, dass sie auch Lösungen gut bewertet bewertet, die durch Umschläge eigentlich völlig entwertet sind. Es ist also besser, die Abstände sämtlicher Knoten zu vergleichen, basieren auf den tatsächlichen und den berechneten Positionen. Dieser muss noch normiert werden. Dafür wird der Ausdruck „Average Relative Deviation“ eingeführt. Die Gleichung lautet:

$$ARD = \frac{2}{n(n-1)} \sum_{i<j} \frac{d_{ij} - \ell_{ij}}{\min(d_{ij} - \ell_{ij})} \tag{4.1}$$

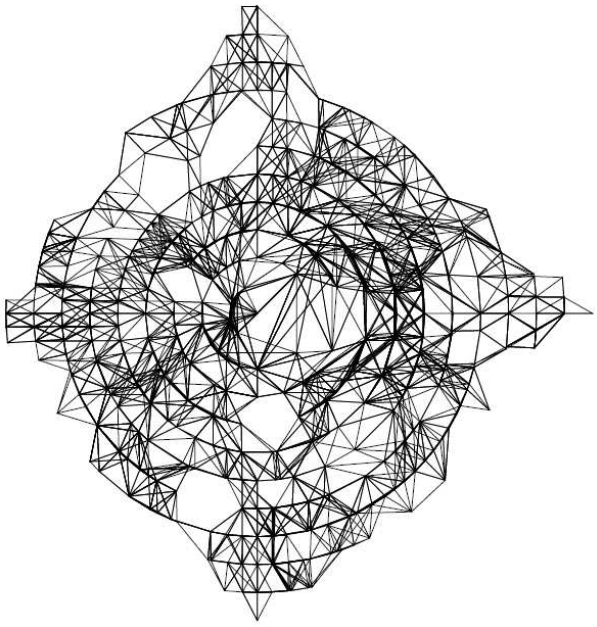


Abbildung 4.1: Ein initiales Layout nach dem AFL-Algorithmus. Man erkennt gut die Struktur der nach Polarkoordinaten angeordneten Knoten. Die Abbildung ist entnommen aus [4].

4.3 Experimentelle Ergebnisse

Für die Simulationen in [4] wurden 1000 Sensoren zufällig im Einheitsquadrat verteilt. Für verschiedene Werte des Kommunikationsradius R und einem Messfehler mit Standardabweichung δ ergeben sich folgende Werte. Es werden dabei im Wesentlichen die verschiedenen Verfahren zur Bestimmung des initialen Layouts verglichen: zufällig (RND), AFL, Spektral-Layout (EIGEN).

	$\delta = 0$			$\delta = 0.05$		
	RND	AFL	EIGEN	RND	AFL	EIGEN
$R = 0.5$	13.5	11.16	0.0780	13.0	10.9	0.0780
$R = 0.6$	11.7	7.38	0.0009	11.7	7.47	0.0110
$R = 0.7$	10.1	5.77	0.0029	10.1	6.11	0.0048
$R = 0.8$	8.75	4.93	0.0018	9.04	4.89	0.0340
$R = 0.9$	7.40	4.27	0.0010	7.56	4.05	0.0270
$R = 1.0$	6.61	3.58	0.0008	6.66	3.45	0.0250

Ausschnitt der experimentellen Ergebnisse aus [4].

4.4 Fazit

Die Messungen zeigen, dass der Algorithmus von Gotsman und Koren um mehrere Größenordnungen besser ist als alles bisher dagewesene. Er erreicht eine so hohe Genauigkeit, dass man das Problem zum ersten Mal als gut gelöst betrachten kann. Die vorherigen Lösungsansätze hatten so große Fehlerbereiche, dass von einer Abbildung der Realität nicht die Rede sein konnte.

Die Autoren sehen jedoch auch noch Spielraum für Verbesserungen. So könnte ihrer Meinung nach für die Formel zur Abbildung des gemessenen Abstands auf das Kantengewicht in Matrix W noch etwas besseres gefunden werden.

Literaturverzeichnis

[1] Ingwer Borg; Patrick Groenen. *Modern Multidimensional Scaling - Theory and Applications*. Springer Series in Statistics. Springer Verlag, 1997.

[2] K. M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17:219–229, 1970.

[3] Bruce Hendrickson. Conditions for unique graph realizations. *SIAM Journal on Computing*, 21(1):65–84, February 1992.

[4] Craig Gotsman; Yehuda Koren. Distributed graph layout for sensor networks. In *12th International Symposium on Graph Drawing*, September 2004.

[5] Yehuda Koren. *On Spectral Graph Drawing*, volume 2697 of *Lecture Notes in Computer Science*, pages 496–508. Springer Verlag, 2003.

[6] Gene H. Golub; Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd ed. edition, 1996.

[7] J. B. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. *Proc. 17th Allerton Conf. in Communications, Control, and Computing*, pages 480–489, 1979.

[8] Nissanka B. Priyantha; Hari Balakrishnan; Erik Demaine; Seth Teller. Anchor-free distributed localization in sensor networks. Technical Report 392, MIT Laboratory for Computer Science, April 2003.