Test

September 26, 2020

Contents

Abstract

al

1 Global Nearest Neighbour: Algorithmus und Idee

1.1 Grundlage GNN

Da die Verfolgung der exakten gesuchten posteorie Wahrscheinlichkeit, aufgrund begrenzter Rechnerzeit und hohen Rechneraufwand, nicht genau bestimmbar ist, sind Approximationen für eine durchführbare Rechnung erforderlich. In diesem Kapitel wird der einfachste und rechnen günstigste Tracking Algorithmus vorgestellt, der Global Nearest Neighbour (GNN).
Wie bereits erwähnt, werden hauptsächlich zwei Annahmen getroffen, um die Problematik noch berechenbar zu gestalten, die sogenannte Merging und Pruning. Der GNN baut sich auf ein abruptes Pruning.

1.1.1 Bekannte Anzahl von Objekten

Eine wichtige Annahme und gleichzeitig Einschränkung dieses Algorithmus, bezieht sich auf die Voraussetzung, dass die Anzahl der Objekten in jedem Zeitschritt n_k bekannt ist. Diese Annahme ist hinsichtlich einer Performance Analyse auf künstliche Testdaten ausreichend, allerdings wird der Algorithmus für eine Realitätsnahe Simulation so gut wie unbrauchbar. In den späteren Kapiteln, wird die *M/N* Methode vorgestellt, die eine Einschätzung von n_k liefern kann, daher wäre der GNN bei einer praktischen Anwendung verwendbar.

1.1.2 Pruning in dem GNN Context

Hinsichtlich der Übersichtlichkeit der vorhanden Problematik, stellt man sich folgendes Szenario vor: Bei jedem Zeitschritt k (immer wenn ein neues Bild aufgenommen wird), erhält der Algorithmus verschiedenen Messungen $\underline{M}_k$ mit x und y Koordinaten und Länge m_k aus der Objektdetektion. Dies enthält sowohl Clutters wie die Koordinaten der echten Objekten o_k. Ohne eine genauere Analyse , sind diese beiden Größen nicht unterscheidbar, daher bewertet man über Zeitschritten hinweg, welche der Koordinaten näherungsweise einem *Constant Velocity Model* entsprechen können. Je mehr die Reihe der Koordinaten einer konstanten Geschwindigkeit Annahme folgt, desto wahrscheinlicher ist es, dass es sich um ein echtes Objekt handelt. Auf der anderen Seite, Reihen von Koordinaten deren vorkommen zufällig erscheinen, werden mit höherer Wahrscheinlich den Clutters zugewiesen.
Eine Datenanalyse ohne das Pruning, müsste alle mögliche Kombinationen der Daten bewerten. Zum Beispiel für lediglich zwei Zeitschritten $k = 0$ und $k = 1$ mit jeweiligen zwei Messungen $m_0 = 2$ und $m_1 = 2$, entstehen 4 Assoziationsmöglichkeiten, da sowohl die erste, wie die zweite Koordinate von der Messung M_0 zwei Alternativen besitzen sich mit den Koordinaten der M_1 zu Assoziieren.
Laut (?, ?) die Gleichung, die die gesamte Anzahl aller Assoziationen N_A für beliebige Objektanzahl n_k und beliebige Messungen m_k vorgibt lautet:

$$N_A(m_k, n_k) = \sum_{\delta=0}^{min(m_k,n_k)} \frac{m_k! n_k!}{\delta!(m_k - \delta)!(n_k - \delta)!} \tag{1}$$

Infolge dessen, für $m_k = n_k = 8$, entstehen bereits 1441729 mögliche Assoziationen. Dieses Beispiel vermittelt wie wesentlich für den Rechneraufwand es sei, die Hypothesen Anzahl mithilfe des Prunings einzuschränken.
Bei dem GNN, wird lediglich die wahrscheinlichste Assoziation beibehalten, alle anderen möglichen Kombinationen werden bei jedem Zeitschritt *weggeprunt*. Diese Eigenschaft verleiht dem GNN die Schnelligkeit und Einfachheit, die seine Anwendung auf kosten der Genauigkeit gerechtfertigten. Aus dem Grund, dass nur die wahrscheinlichste Hypothese bei jedem Zeitschritt k berücksichtigt wird, wirde dieser Algorithmus als *Greedy-Algorithm* bezeichnet.
Mithilfe des Prunings, kann man die folgenden posteriori Wahrscheinlichkeit der Objektzuständen $\underline{x_k}$ darstellen.

$$p_{k|k}^{GNN}(\underline{x_k}) = p_{k|k}^{\theta^*_{1:k}}(\underline{x_k}), \tag{2}$$

wo θ_k^* die wahrscheinlichste Assoziationshypothese repräsentiert. $\theta_{1:k}^*$ stellt eine Reihe von wahrscheinlichsten Assoziationshypothesen bis den Zeitschritt k dar. Das Ziel wäre natürlich die posteriori Wahrscheinlichkeit für die jeweilige Objekte getrennt auszurechnen, also alle $p_{k|k}^{\theta^{*,i}_{1:k}}(\underline{x}_k^i)$ für $i = 1, ..., n_k$.

1.1.3 Allgemeine Vorgehensweise

Mit einer bekannten Anzahl von Objekten n_k und ohne weiteren Vereinfachungen sind die Hauptschritte dieses Algorithmus folgendermaßen abgebildet.
Für jeden Zeitschritt k:

Step 1: **Prädiktion**
Für jedes Objekt i, $p_{k|k-1}^i(\underline{x}_k^i)$ mithilfe einer Chapman-Kolmogorov Prädikation ausrechnen

Step 2: Wahrscheinlichste Hypothese ermitteln.

Step 3: **Update**
Für jedes Objekt i, $p_{k|k}^i(\underline{x}_k^i)$ mithilfe eines Bayes-Update ausrechnen, falls das Objekt detektiert worden ist. Ansonsten $p_{k|k}^i(\underline{x}_k^i) = (1-P^D)p_{k|k-1}^i(\underline{x}_k^i)$.
$P^D \in [0, 1]$ sei die Detektionsrate und wird als Konstant angenommen.

Step 4: Zustand aus der Aposteriori Wahrscheinlichkeit mithilfe des Erwartungswerts bestimmen
$\underline{x}_k = \int x_k^i p_{k|k}^{GNN}(\underline{x_k}) dx_k^i$

1.1.4 Unabhängige Objektzustände

Da diese Schritte für eine reale Anwendung unpraktisch sein können, setzt man folgenden Annahmen voraus.

Assumption 1 - *Das Zustandsmodell sei linear.*

Assumption 2 - *Die Objektzustände seien linear unabhängig.*

Assumption 3 - *Gauß'sche Wahrscheinlichkeitsdichte der Objektbewegung.*

Seien Sie erfüllt, ist der Einsatz eines Kalman-Filters bei der Prädikation beziehungsweise des Updates möglich, was die Berechnung wesentlich vereinfacht. Außerdem, kann man sich auf die Berechnung auf Step 4: verzichten, da eine direkte Ermittelung der Zustände ohne die Aposteriori Wahrscheinlichkeit möglich ist. Das nächste Kapital zeigt wie der Algorithmus mithilfe dieser Vereinfachungen aufgebaut wird.

1.2 GNN Algorithmus

1.2.1 GNN Kalman-Filter

Wenn die Voraussetzungen 1 bis 3 des vorherigen Kapitels erfüllt sind, kann der GNN die Eingenschaften des Kalman-Filters ausnutzen. Dieses Filter handelt sich um einen Zustandsschätzer, der das Eingangs beziehungsweise Ausgangsrauschen eines dynamischen Systems minimieren kann. Dafür werden zwei Hauptschritten erforderlich: die Prädikation und die Korrektur (oder Update). Das Kalman-Filter benötigt drei Parameter, die die Leistung der Schätzung stark beeinflussen können. Die positiv-definite Matrix der Varianz des Modellrauschens $\mathbf{Q} \in \Re^{4x4}$, die positiv-definite Matrix der Varianz des Messrauschens $\mathbf{R} \in \Re^{2x2}$ und letztlich die symmetrische Objektkovarianzmatrix des Systemrauschens $\mathbf{P}^i \in \Re^{2x2}$. Die Matrizen $\mathbf{Q}$ und $\mathbf{R}$ sind für alle Objekte konstant und sind Zeitinvariant. Matrix $\mathbf{P}^i$ wird bei jedem Korrekturschritt aktualisiert und ist für die jeweilige Objekte i unterschiedlich. Die Prädiktion und Korrektur werden bei jedem Zeitschritt erneut aufgerufen und sind für jedes Objekt $i = 1, ..., n_k$ getrennt auszuführen. Der Einfluss der Wahl der Matrizen $\mathbf{Q}$ und $\mathbf{R}$ sowie des Startwertes der Matrix $\mathbf{P}^i$ wird im Kapitel 1.3 untersucht.

1.2.2 Kalman-Prädiktion

Das Ziel der Prädiktion ist die vorhersage des Zustands des aktuellen Zeitschrittes nur anhand des Modells (Matrizen $\mathbf{F}$ und $\mathbf{H}$) und der Messung der Koordinaten bei dem vorherigen Schritt $k-1$. In diesem Vorgang wird die Kenntnis der Größen des aktuellen Zeitschritts noch nicht vorausgesetzt. Die Prädiktion erfolgt mithilhe folgender Gleichungen.

$$\underline{x}^i_{k|k-1} = F\underline{x}^i_{k-1|k-1}, \tag{3}$$

$$\mathbf{P}^i_{k|k-1} = \mathbf{F}\mathbf{P}^i_{k-1|k-1}\mathbf{F}^T + \mathbf{Q}. \tag{4}$$

Die Notation $[.]_{k|k-1}$ repräsentiert die Vorhersage des Zustands in dem Zeitschritt k gegeben die Informationen aus $k-1$.
Die Prognose der aktuellen Werten ohne das Wissen der eingentlichen Größen verursacht eine Erhöhung der Fehleranfälligkeit, die ohne die Korrektur, die echte Ergebnis über viele Zeitschritten immer weiter verfälscht. Um dies zu vermindern, korrigiert der Update-schritt die vorhergesagte Größe bezüglich der gemessenen Werten.

1.2.3 Datenassoziation und Kostenmatrix

Bevor eine Korrektur auf die Prädiktion angewandt werden kann, wird die Ermittlung der wahrscheinlichsten Hypothese erforderlich. Das heißt: Bei einem Zeitschritt k, erhält man aus der Detektion m_k Koordinaten und aus dem M/N

Algorithmus n_k Objekte. Das Ziel ist natürlich die Objekte einer der m_k Koordinaten zuzuweisen, um eine Korrektur mit den echten Messungen zu ermöglichen. Die Koordinaten die keinem Objekt entsprechen werden als Clutter angesehen. Außerdem, es besteht auch die Möglichkeit, dass ein bestimmtes Objekt nicht detektiert wurde. Um dies zu Lösen verwendet man die Eigenschaften eines *Zuweisungsproblems*.
Zur Veranschaulichung der Problematik, wird folgendes Beispiel präsentiert. An einem bestimmten k sei $n_k = 2$ und $m_k = 3$. Sei $l_{i,j}$ der Kosten, dass Objekt i der Messung j zugewiesen wird und sei $l_{i,0}$ der Kosten für den Fall, dass i nicht detektiert wurde. Man bilde also die Kostenmatrix $\mathbf{L} \in \Re^{n_k \times (m_k + n_k)}$.

$$\mathbf{L} = \begin{bmatrix} l_{1,1} & l_{1,2} & l_{1,3} & l_{1,0} & \infty \\ l_{2,1} & l_{2,2} & l_{2,3} & \infty & l_{2,0} \end{bmatrix}$$

Die Koordinaten eines Objekts können entweder genau einer Messung entsprechen, oder nicht detektiert werden. Eine Messung kann maximal einem Objekt Zugewiesen werden. Die "∞" Einträge der Matrix schließen sinnlose Assoziationen aus. Für die Lösung des Problems, suggeriert (?, ?) die Anwendung eines *Hungarian-Algorithmus*, der das folgende Optimierungsproblem erfasst:

$$\begin{aligned} &\underset{\mathbf{A}}{\text{minimiere}} \quad tr(\mathbf{A}^T\mathbf{L}) \\ &\text{u.d.v} \qquad\quad \sum_j \mathbf{A}_{i,j} = 1, \quad \sum_i \mathbf{A}_{i,j} = 1. \end{aligned} \tag{5}$$

Matrix $\mathbf{A}$ enthält lediglich die Werte 0 und 1 und die Summe ihrer Spalten und Zeilen ergeben immer genau 1. Die Anweisung $tr(.)$ repräsentiert die Spur einer Matrix und daher die Operation $tr(\mathbf{A}^T\mathbf{L})$ ergibt die Assoziationskosten einer bestimmten Zuordnung.
Zur Bestimmung der Kostenmatrix Elementen $l_{i,j}$ stellt (?, ?), unter der Voraussetzung eines linearen und gauß'schen Modells und einer konstanten P^D, die untere Gleichungen vor.

$$l_{i,0} = -ln(1 - P^D) \tag{6}$$

$$l_{i,j} = -[ln(\frac{P^D}{\lambda_c}) - 0,5log(det(2\pi\mathbf{S}_i)) - 0.5(\underline{z}_j - \underline{\tilde{z}}_i)\mathbf{S}_i^{-1}(\underline{z}_j - \underline{\tilde{z}}_i)],$$

wo die Clutter-Intensität $\lambda_c = 1 - \dfrac{n_k}{m_k}$, die Koordinaten des Objektes i $\underline{\tilde{z}}_i = \mathbf{H}\underline{x}_{k|k-1}$ und die Kovarianzinovationsmatrix $\mathbf{S}_i = \mathbf{H}\mathbf{P}^i_{k|k-1}\mathbf{H}^T + \mathbf{R}$ die restlichen Variablen darstellen.
Die Elemente $l_{i,0}$ bilden die Gewichte ab, für den Fall dass i nicht detektiert wurde. Die Elemente $l_{i,j}$ sind, wie bereits erwähnt, die Gewichte, wenn Objekt i der Messung j zugewiesen wird.
Die Lösung dieses Assoziationsproblem wird mithilfe eines Python-Pakets, welches von (?, ?) zur Verfügung gestellt worden ist.
Zu den Gunsten der Erhaltung einer kompakteren Schreibweise der optimalen Assoziation, führt man die Variable $\theta \in \Re^{n_k}$ ein. $\theta_i = 0$ bedeutet, dass keine neue Detektion für das Objekt i vorhanden ist und $\theta_i = j$ gibt an, dass i der Messung j zugeordnet wurde.

1.2.4 Kalman-Update

Nach der Ermittlung der wahrscheinlichste Hypothese, wird eine Korrektur benötigt, um die Fehlervarianz des Schätzers zu reduzieren. Die Korrektur wird mithilfe der Messung der Koordinaten von was als ein wahrscheinliches Objekt betrachtet wird. Folgende Gleichungen repräsentieren diesen Vorgang.

$$\mathbf{K}^i = \mathbf{P}^i_{k|k-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}^i_{k|k-1}\mathbf{H}^T + \mathbf{R})^{-1}. \tag{7}$$

$\mathbf{K}^i$ sei die Kalman-Verstärkung und $\mathbf{P}_{k|k-1}$ die Kovarianzmatrix aus dem Prädiktionsschritt. Letztlich lässt sich den aktuellen Zustand $\underline{x}^i_{k|k}$ beziehungsweise die neue Kovarianzmatrix $\mathbf{P}^i_{k|k-1}$ folgendermaßen berechnen.

$$\underline{x}^i_{k|k} = \begin{cases} \underline{x}^i_{k|k-1} + \mathbf{K}^i(\underline{z}^{\theta^{*,i}} - \mathbf{H}\underline{x}^i_{k|k-1}) & \text{für } \theta^{*,i} \neq 0 \\ \underline{x}^i_{k|k-1} & \text{für } \theta^{*,i} = 0, \end{cases} \tag{8}$$

$$\mathbf{P}^i_{k|k} = \begin{cases} \mathbf{P}^i_{k|k-1} - \mathbf{K}^i\mathbf{H}\mathbf{P}^i_{k|k-1} & \text{für } \theta^{*,i} \neq 0 \\ \mathbf{P}^i_{k|k-1} & \text{für } \theta^{*,i} = 0. \end{cases}$$

Der Vektor $\underline{z}^{\theta^{*,i}}$ stellt die gemessenen Koordinaten dar, die als wahrscheinlichste Fortsetzung der Bewegung des Objekten i angesehen wird. Ein $\theta^{*,i} \neq 0$ bedeutet, dass das Objekt i detektiert worden ist. Wenn $\theta^{*,i} = 0$ (keine Detektion des Objekts), ist keine Korrektur möglich und daher werden die Werte der Prädiktion weitergereicht.

1.2.5 M/N Algorithmus

1.2.6 Startwerte

1.2.7 Deaths and Births

1.2.8 GNN Zusammenfassung

1.3 Ergebnisauswertung auf Testdaten

1.3.1 Einfluss der Hyperparametern

1.4 Einschränkung des Algorithmus

zitiere (?, ?), und (?, ?)