1

Constraint Based Object State Modeling

Daniel Göhring, Heinrich Mellmann, Hans-Dieter Burkhard

Institut für Informatik
LFG Künstliche Intelligenz
Humboldt-Universität zu Berlin
Unter den Linden 6
10099 Berlin, Germany
http://www.aiboteamhumboldt.com

Summary. Modeling the environment is crucial for a mobile robot. Common approaches use Bayesian filters like particle filters, Kalman filters and their extended forms. We present an alternative and supplementing approach using constraint techniques based on spatial constraints between object positions. This yields several advantages: a) the agent can choose from a variety of belief functions, b) the computational complexity is decreased by efficient algorithms. The focus of the paper are constraint propagation techniques under the special requirements of navigation tasks.

1.1 Introduction

Modeling the world state is important for many robot tasks. But usually robots have a limited field of view, which makes it hard to acquire the whole surrounding from one image. Bayesian filters [1] have been very successful in solving this problem by incorporating sensor data over time. A very famous member of the Bayesian filter family is the Kalman filter [2] using Gaussian distribution functions. But many distributions can neither be processed by a Kalman filter nor by one of its extensions. For non gaussian distributions particle filters have become very popular. But the calculation of the sample set can become very costly, making it inappropriate for real time applications.

Given an image of a scene, we have constraints between the objects in the image and the objects in the scene. Object parameters, image parameters and camera parameters are dependent by related constraints. Given odometry (or control) data, subsequent positions are constrained by measured speed and direction of movements. They can be combined with sensor measurements [3].

We have to deal with incomplete or with noisy measurements. With incomplete measurements, the result of constraint propagation will be ambiguous, while noisy measurements may lead to inconsistent constraints. Related qual-

ity measures have been discussed in our paper [4]. In this paper we discuss constraint propagation methods for solving navigation problems.

The main difference to classical propagation is due to the fact that navigation tasks do always have a solution in reality. For that, inconsistencies have to be resolved e.g. by relaxing constraints. Moreover, navigation tasks are not looking for a single solution of the constraint problem. Instead, all possible solutions are interesting in order to know about the ambiguity of the solution (which is only incompletely addressed by particle filters). For that, the notion of conservative propagation functions is introduced. It can be shown that this notion coincides to some extend to the classical notion of local consistency (for maximal locally consistent intervals).

The paper is structured as follows: Section 1.2 gives an introduction and an example for description and usage of constraints generated from sensor data. In Section 1.3 we present the formal definitions and the backgrounds for usage of constraints. Basics of constraint propagation in the context of navigation tasks are discussed in Section 1.4, and an efficient algorithm is presented.

1.2 Perceptual Constraints

A constraint C is defined over a set of variables $v(1), v(2), ..., v(k)$. It defines the values those variables can take:

$$C \subseteq Dom(v(1)) \times ... \times Dom(v(k))$$

We start with an example from RoboCup where the camera image of a robot shows a goal in front and the ball before the white line of the penalty area (Figure 1.1). It is not too difficult for a human interpreter to give an estimate for the position (x_B, y_B) of the ball and the position (x_R, y_R) of the observing robot. Humans can do that, regarding relations between objects, like the estimated distance d_{BR} between the robot and the ball, and by their knowledge about the world, like the positions of the goalposts and of the penalty line.

The program of the robot can use the related features using image processing. The distance d_{BR} can be calculated from the size of the ball in the image, or from the angle of the camera. The distance d_{BL} between the ball and the penalty line can be calculated, too. Other values are known parameters of the environment: $(x_{Gl}, y_{Gl}), (x_{Gr}, y_{Gr})$ are the coordinates of the goalposts, and the penalty line is given as the set of points $\{(x, b_{PL}) | -a_{PL} \leq x \leq a_{PL}\}$. The coordinate system has its origins at the center point, the y-axis points to the observed goal.

The relations between the objects can be described by constraints. The following four constraints are obvious by looking to the image, and they can be determined by the program of the observing robot:

C_1: The view angle γ between the goalposts (the distance between them in the image) defines a circle (periphery circle).

C_2: The ball lies in the distance d_{BL} before the penalty line.
C_3: The distance d_{BR} between the robot and the ball defines a circle such that the robot is on that circle around the ball.
C_4: The observer, the ball and the left goal post are on a line.

The points satisfying the constraints by C_1 (for the robot) and by C_2 (for the ball) can be visualized immediately on the playground as in Figure 1.1.

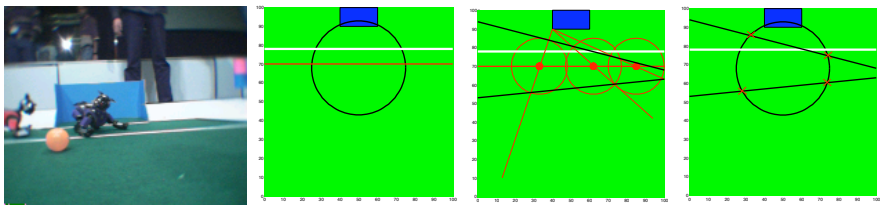

Fig. 1.1. Example from RoboCup (Four legged league): A robot is seeing a goal and the ball before the penalty line. The first Figure on left side illustrates the scene from the view of the robot. The next three figures show the related constraints that can be used for localization. Left: The picture shows a part of the field with the goal and the white penalty line, the periphery circle according to C_1, and the line of the Ball-Line-Constraint C_2. Middle: The picture shows the Constraint C_2 for the ball, some of the circles according to constraint C_5, some of the lines according to C_4, and the resulting two lines for C_6. Right: Constraints according to C_7: The position of the robot is one of the four intersection points between the periphery circle (C_1) and the lines according to C_6.

The constraint by C_3 does not give any restriction to the position of the ball. The ball may be at any position on the playground, and then the robot has a position somewhere on the circle around the ball. Or vice versa for reasons of symmetry: The robot is on any position of the playground, and the ball around him on a circle. In fact, we have four variables which are restricted by C_3 to a subset of a four dimensional space. The same applies to constraint C_4.

The solution (i.e. the positions) must satisfy all four constraints. We can consider all constraints in the four dimensional space of the variables (x_B, y_B, x_R, y_R) such that each constraint defines a subset of this space. Then we get the following constraints:

$$C_1 = \{(x_B, y_B, x_R, y_R) | \arctan \frac{y_{Gl} - y_R}{x_{Gl} - x_R} - \arctan \frac{y_{Gr} - y_R}{x_{Gr} - x_R} = \gamma\} \quad (1.1)$$

$$C_2 = \{(x_B, y_B, x_R, y_R) | (x_B \in [-a_{PL}, a_{PL}], y_B = b_{PL} - d_{BL}\} \quad (1.2)$$

$$C_3 = \{(x_B, y_B, x_R, y_R) | (x_B - x_R)^2 + (y_B - y_R)^2 = d_{BR}^2\} \quad (1.3)$$

$$C_4 = \{(x_B, y_B, x_R, y_R) | \frac{x_R - x_B}{y_R - y_B} = \frac{x_B - x_{Gl}}{y_B - y_{Gl}}\} \quad (1.4)$$

Then the possible solutions (as far as determined by C_1 to C_4) are given by the intersection $\bigcap_{1,...,4} C_i$. According to this fact, we can consider more constraints $C_5, \ldots, C_n$ as far as they do not change this intersection, i.e. as far as $\bigcap_{1,...,n} C_i = \bigcap_{1,...,4} C_i$. Especially, we can combine some of the given constraints.

By combining C_2 and C_3 we get the constraint $C_5 = C_2 \cap C_3$ where the ball position is restricted to any position on the penalty line, and the player is located on a circle around the ball. Then, by combining C_4 and C_5 we get the constraint $C_6 = C_4 \cap C_5$ which restricts the positions of the robot to the two lines shown in Figure 1.1 (middle).

Now intersecting C_1 and C_6 we get the constraint C_7 with four intersection points as shown in Figure 1.1 (right). According to the original constraints C_1 to C_4, these four points are determined as possible positions of the robot. The corresponding ball positions are then given by C_2 and C_4.

1.3 Formal Definitions of Constraints

We define all constraints over the set of all variables $v(1), v(2), ..., v(k)$ (even if some of the variables are not affected by a constraint). The domain of a variable v is denoted by $Dom(v)$, and the whole universe under consideration is given by

$$U = Dom(v(1)) \times \cdots \times Dom(v(k))$$

For this paper, we will consider all domains $Dom(v)$ as (may be infinite) intervals of real numbers, i.e. $U \subseteq \mathbb{R}^k$.

Definition 1. *(Constraints)*

1. *A* **constraint** C *over* $v(1), ..., v(k)$ *is a subset* $C \subseteq U$*.*
2. *An assignment* β *of values to the variables* $v(1), ..., v(k)$*, i.e.* $\beta \in U$*, is a* **solution** *of* C *iff* $\beta \in C$*.*

Definition 2. *(Constraint Sets)*

1. *A* **constraint set** $\mathcal{C}$ *over* $v(1), ..., v(k)$*is a finite set of constraints over those variables:* $\mathcal{C} = \{C_1, ..., C_n\}$*.*
2. *An assignment* $\beta \in U$ *is a* **solution** *of* $\mathcal{C}$ *if* β *is a solution of all* $C \in \mathcal{C}$*, i.e. if* $\beta \in \bigcap \mathcal{C}$
3. *A constraint set* $\mathcal{C}$ *is* **inconsistent** *if there is no solution, i.e. if* $\bigcap \mathcal{C} = \emptyset$

The problem of finding solutions is usually denoted as solving a constraint satisfaction problem (CSP) which is given by a constraint set $\mathcal{C}$. By our definition, a solution is a point of the universe U, i.e. an assignment of values to all variables. For navigation problems it might be possible that only some variables are of interest. This would be the case if we are interested only in the position of the robot in our example above. Nevertheless we had to solve the whole problem to find a solution.

In case of robot navigation, there is always a unique solution of the problem in reality (the positions in the real scene). This has an impact on the interpretation of solutions and inconsistencies of the constraint system (cf. Section 1.4).

The constraints are models of relations (restrictions) between objects in the scene. The information is derived from sensory data, from communication with other robots, or from knowledge about the world – as in the example from above. Since information may be noisy, the constraints may not be as strict as in the introductory example from Section 1.2. Instead of a circle we get an annulus for the positions of the robot around the ball according to C_3 in the example. In general, a constraint may concern a subspace of any dimension (e.g. the whole penalty area, the possible positions of an occluded object, etc.). Moreover, constraints need not to be connected: If there are indistinguishable landmarks, then the distance to such landmarks defines a constraint consisting of several circles.

Other constraints are given by velocities: Changes of locations are restricted by the direction and speed of objects. This means that a position cannot change too much within a short time.

There are many redundancies which are due to all available constraints. Visual information in images usually contain lots of useful information: Size and appearance of observed objects, bearing angles, distances and other relations between observed objects, etc. Only a very small part of this information is usually used in classical localization algorithms. This might have originated in the fact, that these algorithms have been developed for range measurements. Another problem is the large amount of necessary calculation for Bayesian methods (grids, particles). Kalman filters can process such large amounts, but they rely on additional presumptions according to the underlying statistics.

Like Kalman filters, the constraint approach has the advantage, that it can simultaneously compute positions of different objects and the relations between them. Particle filters can deal only with small dimensions of search spaces.

For constraint methods, we have the problem of inconsistencies. According to the noise of measurements, it may be impossible to find a position which is consistent with all constraints. In our formalism the intersection of all constraints will be empty in such a case. Inconsistency in constraint satisfaction problems means usually that there does not exist a solution in reality. But in our situation, the robot (and the other objects) do have their coordinates, only the sensor noise corrupted the data. Related quality measures for constraint sets have been investigated in [4].

1.4 Constraint Propagation

Known techniques (cf. e.g. [5] [6]) for constraint problems produce successively reduced sets leading to a sequence of decreasing restrictions

$$U = D_0 \supseteq D_1 \supseteq D_2, \supseteq \dots$$

Restrictions for numerical constraints are often considered in the form of k-dimensional intervals $I = [\mathbf{a}, \mathbf{b}] := \{\mathbf{x} | \mathbf{a} \leq \mathbf{x} \leq \mathbf{b}\}$ where $\mathbf{a}, \mathbf{b} \in \mathbf{U}$ and the $\leq$-relation is defined componentwise. The set of all intervals in U is denoted by $\mathcal{I}$. A basic scheme for constraint propagation with

- A constraint set $\mathcal{C} = \{C_1, ..., C_n\}$ over variables $v(1), ..., v(k)$ with domain $U = Dom(v(1)) \times ... \times Dom(v(k))$.
- A selection function $c : \mathbb{N} \to \mathcal{C}$ which selects a constraint C for processing in each step i.
- A propagation function $d : 2^U \times \mathcal{C} \to 2^U$ for constraint propagation which is monotonously decreasing in the first argument: $d(D, C) \subseteq D$.
- A stop function $t : \mathbb{N} \to \{true, false\}$.

works as follows:

Definition 3. *(Basic Scheme for Constraint Propagation, BSCP)*

Step(0) Initialization: $D_0 := U$, $i := 1$
Step(i) Propagation: $D_i := d(D_{i-1}, c(i))$.
If $t(i) = true$*: Stop.*
Otherwise $i := i + 1$*, continue with Step(i).*

We call any algorithm which is defined accordingly to this scheme a BSCP-algorithm.

The restrictions are used to shrink the search space for possible solutions. If the shrinkage is too strong, possible solutions may be lost. For that, backtracking is allowed in related algorithms.

To keep the scheme simple, the functions c and t depend only on the time step. A basic strategy for c is a round robin over all constraints from $\mathcal{C}$, while more elaborate algorithms use some heuristics. A more sophisticated stop criterion t considers the changes in the sets D_i. Note that the sequence needs not to become stationary if only $D_i = D_{i-1}$. Actually, the sequence $D_0, D_1, D_2, \dots$ needs not to become stationary at all.

For localization problems with simple constraints it is possible to compute the solution directly:

Corollary 1. *If the propagation function d is defined by $d(D, C) := D \cap C$ for all $D \subseteq U$ and all $C \in \mathcal{C}$, then the sequence becomes stationary after $n = card(\mathcal{C})$ steps with the correct result $D_n = \bigcap \mathcal{C}$.*

For simpler calculations, the restrictions D_i are often taken in simpler forms (e.g. as intervals) and the restriction function d is defined accordingly.

Usually constraint satisfaction problems need only some but not necessarily all solutions. For that, the restriction function d does not need to regard all possible solutions (i.e. it need not be conservative according to definition 5 below). A commonly used condition is local consistency:

Definition 4. *(Locally consistent propagation function)*

1. *A restriction D is called* **locally consistent w.r.t. a constraint** *C if*

$$\forall d = [d_1, ..., d_k] \in D \quad \forall i = 1, ..., k \quad \exists d' = [d'_1, ..., d'_k] \in D \cap C \ : d_i = d'_i$$

   i.e. if each value of a variable of an assignment from D can be completed to an assignment in D which satisfies C.
2. *A propagation function $d : 2^U \times \mathcal{C} \to 2^U$ is* **locally consistent** *if it holds for all D, C: $d(D, C)$ is locally consistent for C.*
3. *The* **maximal locally consistent** *propagation function $d_{maxlc} : 2^U \times \mathcal{C} \to 2^U$ is defined by $d_{maxlc}(D, C) := Max\{d(D, C) | d \text{ is locally consistent}\}$.*

Since the search for solutions is easier in a more restricted the search space (as provided by smaller restrictions D_i), constraint propagation is often performed not with d_{maxlc}, but with more restrictive ones. Backtracking to other restrictions is used if no solution is found.

For localization tasks, the situations is different: We want to have an overview about all possible poses. Furthermore, if a classical constraint problem is inconsistent, then the problem has no solution. In localization problem, there does exist a solution in reality (the real poses of the objects under consideration). The inconsistency is caused e.g. by noisy sensory data. For that, some constraints must be relaxed or enlarged in the case of inconsistencies. This can be done during the propagation process by the choice of even a larger restrictions than given by the maximal locally consistent restriction function.

Definition 5. *(Conservative propagation function)*
A propagation function $d : 2^U \times \mathcal{C} \to 2^U$ is called **conservative** *if $D \cap C \subseteq d(D, C)$ for all D and C.*

Note that the maximal locally consistent restriction function d_{maxlc} is conservative. We have:

Proposition 1. *Let the propagation function d be conservative.*

1. *Then it holds for all restrictions D_i : $\bigcap \mathcal{C} \subseteq D_i$.*
2. *If any restriction D_i is empty, then there exists no solution, i.e. $\bigcap \mathcal{C} = \emptyset$.*

If no solution can be found, then the constraint set is inconsistent. There exist different strategies to deal with that:

- enlargement of some constraints from $\mathcal{C}$,
- usage of only some constraints from $\mathcal{C}$,
- computation of the best fitting hypothesis according to $\mathcal{C}$.

We have discussed such possibilities in the paper [4].

As already mentioned above, intervals are often used for the restrictions D, since the computations are much easier. Constraints are intersected with intervals, and the smallest bounding interval can be used as a conservative result. Examples are given in Fig. 1.2.

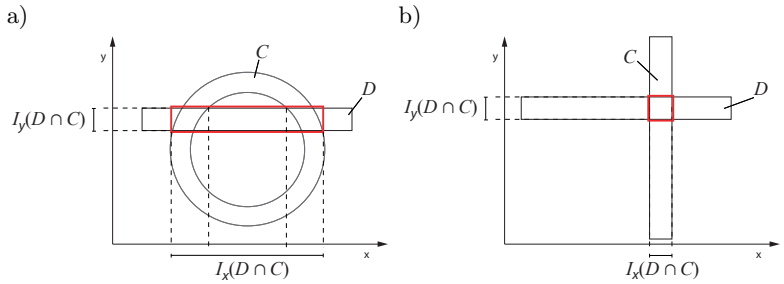

Fig. 1.2. Constraint propagation with intervals D for a) a circular constraint C b) a rectangular constraint C. *Intervals of Projection* w.r.t. $C \cap D$ are illustrated.

Definition 6. *(Interval Propagation)*

1. *A propagation function d is called an* **interval propagation function** *if the values of d are always intervals.*
2. *The* **minimal conservative** *interval propagation function $d_{minc} : 2^U \times \mathcal{C} \rightarrow \mathcal{I}$ is defined by $d_{minc}(D, C) := Min\{I | I \in \mathcal{I} \wedge D \cap C \subseteq I)\}$ for all D and C.*

The results by minimal conservative interval propagation functions can be computed using projections.

Definition 7. *(Interval of projection)*
The (one-dimensional) **Interval of projection w.r.t. to a set** *$M \subseteq U$ for a variable v is defined as the smallest interval containing the projection $\Pi_v(M)$ of M to the variable v: $I_v(M) = Min\{I | I \subseteq \mathbb{R} \wedge \Pi_v(M) \subseteq I\}$. It can be computed as $I = [a, b]$ with $a := Min(\Pi_v(M))$ and $b := Max(\Pi_v(M))$.*

Both, maximal local consistency and minimal conservatism leads to the same results, and both can be computed using the projections (Figure 1.2):

Proposition 2.

1. $d_{maxlc}(D, C) = d_{minc}(D, C)$
2. $d_{minc}(D, C) = I_{v(1)}(D \cap C) \times .... \times I_{v(k)}(D \cap C)$.

While local consistency is the traditional approach (to find only some solutions), the approach with conservative intervals is more suited for localization tasks because it can be modified w.r.t. to enlarging constraints during propagation for preventing from inconsistency. In case of inconsistencies, the algorithm below is modified accordingly in step 6. The related work is still under investigation.

The following simple and practicable algorithm is used for propagation. The stop condition compares the progress after processing each constraint

once. Since stabilization needs not to occur, we provide an additional time limit. Note that the step counting s is not identical to the steps i in the basic scheme BSCP (but could be arranged accordingly).

Algorithm 1: Constraint Propagation with Minimal Conservative Intervals, MCI-algorithm

Input: constraint set $\mathcal{C} = \{C_1, ..., C_n\}$ with variables $\mathcal{V} = \{v_1, ..., v_k\}$ over domain U and a time bound T
Data: $D \leftarrow U$, $s \leftarrow 1$, $D_{old} \leftarrow \emptyset$
Result: minimal conservative k-dimensional interval D

```
1  while s < T & D ≠ D_old do
2  |  D_old ← D;
3  |  foreach C ∈ C do
4  |  |  foreach v ∈ V do
5  |  |  |  D(v) ← I_v(D ∩ C);
6  |  |  end
7  |  |  D ← D(v_1) × ··· × D(v_n);
8  |  end
9  |  s ← s + 1;
10 end
```

1.4.1 Experimental Results

In our experiments within the RoboCup soccer domain (see section 1.2), we compared a standard implementation of a Monte-Carlo particle filter with the algorithm described above.

We used constraints given by fixed objects like goalposts, flags and field lines identified in the images by the camera of the robot. It was easy to derive the related constraints: distances to landmarks are defined by circular rings in a generic form, where only the distances derived from the vision system of the robot have to be injected. Constraints given by observed field lines are defined by rectangles and angles, the distances and the horizontal bearings are sufficient to define these constraints. All this can be done automatically.

While the particle filter used data from odometry, the constraint approach was tested with only the actual vision data. Since we were able to exploit various redundancies for the MCI, the accuracy of the results were comparable.

Our experiments showed that the MCI algorithm works several times faster than a related particle filter. We performed experiments with a different number of particles. Even with very small sample sets (about 50 particles) the computational costs for the MCPF were several times higher than for MCI. A disadvantage of particle based approaches is that many particles are necessary to approximate the belief which comes at high computational costs.

In further experiments we investigated more ambiguous data (i.e. when only few constraints are available). In this case, the MCI provided a good

estimation of all possible positions (all those positions which are consistent with the vision data). The handling of such cases is difficult for MCPF because many particles would be necessary. Related situations may appear for sparse sensor data and for the kidnapped robot problem. Odometry can improve the results in case of sparse data (for MCPF as well as with additional constraints in MCI). But we would argue that the treatment of true ambiguity by MCI is better for the kidnapped robot problem.

1.5 Conclusion

Constraint propagation techniques are an interesting alternative to probabilistic approaches. From a theoretical point of view, they could help for better understanding of navigation tasks at all. For practical applications they permit the investigation of larger search spaces employing the constraints between various data. Therewith, the many redundancies in images can be better used. This paper has shown how sensor data can be transformed into constraints. We presented an algorithm for constraint propagation and discussed some differences to classical constraint solving techniques. In our experiments, the algorithm outperformed classical approaches like particle filters.

The different strategies for dealing with inconsistencies have to be investigated in more detail. This will be done by connecting the results from this paper with our results from [4]. In further work we will analyze constraint based approaches for cooperative object modeling tasks as well as very dynamic situations with quickly changing object states.

References

1. Gutmann, J.S., Burgard, W., Fox, D., Konolige, K.: An experimental comparison of localization methods. In: Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (1998)
2. Kalman, R.: A new approach to linear filtering and prediction problems. Transactions of the ASME - Journal of Basic Engineering **82** (1960) 35–45
3. Jüngel, M.: Memory-based localization. (2007) Proceedings of the CS&P 2007.
4. Göhring, D., Gerasymova, K., Burkhard, H.D.: Constraint based world modeling for autonomous robots. (2007) Proceedings of the CS&P 2007.
5. Davis, E.: Constraint propagation with interval labels. Artificial Intelligence **32** (1987)
6. Goualard, F., Granvilliers, L.: Controlled propagation in continuous numerical constraint networks. ACM Symposium on Applied Computing (2005)