2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Cooperative Anchoring in Heterogeneous Multi-Robot Systems

Kevin LeBlanc and Alessandro Saffiotti

Abstract— **Highly heterogeneous robotic systems are becoming increasingly common, as are robotic systems integrated with smart environments. In such distributed systems, there are many different sources and types of information, which need to be coordinated and combined effectively. The problem of *cooperative anchoring* is (roughly) the problem of, in a distributed system, determining which items of information refer to the same objects, and combining these items accordingly. In this paper, we define a general computational framework for cooperative anchoring inspired by work on conceptual spaces and (single-robot) perceptual anchoring. We also discuss an implementation of this framework which uses tools from fuzzy logic, and we present an illustrative experiment.**

I. Introduction

One of the interesting aspects of systems which use multiple heterogeneous robots, as opposed to single robots, is the richness of available information. In cooperative settings, robots do not need to rely solely on their own perception to obtain information about the environment; they can also receive information from other agents. This information is often of essentially different types. For instance, both perceptual (e.g., data from distributed cameras) and non-perceptual (e.g., symbolic knowledge from a database) information may be available. While this richness of information opens a new landscape of opportunities, it also adds a number of fundamental challenges — namely, those of *representing, communicating, comparing and fusing* this information.

These challenges are particularly important when dealing with robots integrated with smart environments [14], [13], [15]. In smart environments, many devices, including some objects beings observed by robots, can provide information. These devices might not be robots in the traditional sense, since they may not have actuators, or even perception – however, they can be treated as subsets of normal single-robot systems.

To clarify what we mean by "essentially different types of information", consider this scenario. A robot, called Pippi, is told to fetch parcel number 21 from the entrance, where a bunch of parcels are lying on the floor. In order to perform this task, Pippi can rely on several sources of information:

- symbolic information contained in Pippi's task planner:
```
(and (recipient parcel-21 alex)
     (position  parcel-21 entrance))
```
- an RFID reader on the floor able to read information stored in an RFID tag attached to the parcel, like:
```
<xml>
  <refnum>21</refnum>
  <texture>striped</texture>
</xml>
```
- a robot called Astrid, which has a vision system capable of detecting both the color and the texture of the parcel; however due to poor self-localisation, Astrid cannot distinguish its position from that of another parcel:
```
{color(79,6,210),texture(striped)}
{color(2,5,212),texture(none)}
```
- Pippi's on-board camera, which can detect the color and approximate position of the parcel, like:
```
{position(293,392),color(77,7,200)}
```
- a black and white security camera mounted on the ceiling, which can detect the precise position of the parcel, due to its fixed position and bird's eye perspective, like:
```
blob[88] = {position(295,396)}
```

Using the symbolic information in the task planner, the system knows to look near the entrance for the parcel. Using the texture information from the RFID tag, Astrid is able to determine the color of the striped parcel. With this, Pippi can detect the approximate position of the green striped parcel near the entrance. A more precise position estimate can then be obtained from the overhead camera. Current approaches to multi-sensor fusion and to cooperative perception are typically unable to handle such inherently different types of information coming from distributed sources.

In this work we propose a framework which addresses these challenges. The main contribution of the framework is that it allows the different types of information which are present in *heterogeneous, distributed robotic systems* to be effectively matched and fused using the same representation.

Information is represented using geometric spaces, inspired by Gärdenfors's *conceptual spaces* [9]. The framework is inspired by work on *perceptual anchoring* [7], [8], [6]. Perceptual anchoring, or simply anchoring, has been defined as the process of creating and maintaining, in a robotic system, the correspondence between symbols and sensor data referring to the same physical objects. We extend this notion to distributed settings, in which several robots share different types of information about objects.

The rest of this paper is organized as follows. The next section defines the problem of cooperative anchoring, and discusses its relation to other well known problems in robotics. Section III details our computational framework for cooperative anchoring. Section IV describes a sample implementation of this framework based on tools from fuzzy logic. Finally, in section V we present a simple experiment, which illustrates how the framework can be used.

This work is supported in part by CUGS (Swedish computer science graduate school), and in part by ETRI (Electronics and Telecommunications Research Institute, Korea), through the project "Embedded Component Technology and Standardization for URC (2004-2008)".

Both authors are with the AASS Mobile Robotics Lab, Örebro University, Örebro, Sweden, {klc,asaffio}@aass.oru.se

II. Problem Statement

We define the *cooperative anchoring* problem as:

> *the problem of creating and maintaining, in a distributed robot system, the correspondence between items of information which refer to the same physical objects.*

Cooperative anchoring generalizes the single-robot perceptual anchoring problem in two ways. First, we consider *distributed systems*, in which items of information can be produced by and/or stored in different agents. This includes multi-robot scenarios and tasks which are not covered by the single-robot case. Second, we deal with generic *items of information*, instead of focusing only on symbol-based and sensor-based information. When dealing with highly heterogeneous multi-robot systems, many different types of information may be encountered. The first extension calls for mechanisms to adequately combine items of information coming from distributed sources. The second extension calls for a general way to represent, compare and fuse heterogeneous items of information. The approach described in the next sections addresses both issues.

A. Mathematical Formulation

We assume that the environment contains N *objects* of interest denoted by $\{o_1, \ldots, o_N\}$, where the value of N is unknown. We then consider a distributed system consisting of a set of M physically embedded agents, or *robots*, denoted by $\{r_1, \ldots, r_M\}$, with $M > 0$. Note that the word robot is used quite loosely here; for example a fixed camera may also be referred to as a robot. Each robot r_i includes K_i *information sources*, denoted by $\{s^i_1, \ldots, s^i_{K_i}\}$, with $K_i > 0$.

Each source s^i_j produces items of information in some (possibly complex) domain Y^i_j. For instance, a vision system may produce image regions with associated parameters like position, size, color, and so on; a symbolic knowledge base may produce sets of predicates to connotate a given object, like {`green, parcel, urgent`}.

At any sampling time t, each source s^i_j produces a (possibly empty) set of items of information $y^i_j \subseteq Y^i_j$. We assume that each item of information refers to a single object o_h in $\{o_1, \ldots, o_N\}$, although we do not know which one. We also assume that no two items originating from the same information source at the same time t can refer to the same object. These assumptions are normally made in data-association applications. The information available to robot r_i at time t is given by the set $\Phi^i(t) = \bigcup_{j=1,\ldots,K_i} y^i_j$. The information available to the entire distributed system at time t is given by the set $\Phi(t) = \bigcup_{i=1,\ldots,M} \Phi^i(t)$.

Given the above ingredients, the cooperative anchoring problem includes the following three sub-problems.

1) Data association: This is the problem of determining which items of information refer to the same objects. For the entire distributed system, this can be seen as the problem of finding a partitioning of the set $\Phi(t)$ such that each partition contains all and only items which refer to the same object. If a notion of distance, or similarity, between items of information is available, the data association problem can be seen as a clustering problem.

2) Fusion: Once a decision regarding data association has been taken, items which have been determined to refer to the same object can be combined, in order to obtain a new estimate of the object's properties. Done correctly, information fusion can yield improved knowledge about the state of the world. In general, redundant information can be used to reduce uncertainty and imprecision, while complementary information can be used to resolve ambiguities and incompleteness.

3) Prediction: In order to "maintain in time" estimates of object properties, we need to propagate the results of the fusion of information at time t to subsequent points in time. This predicted state can be treated as an extra information source, allowing information about the same objects received at different times to be combined. In a dynamic world, prediction is usually performed using a model which predicts how the properties of objects evolve in time. In this paper we do not address the temporal aspect; instead we focus on the two previous problems, and assume a static world model.

B. Related Work

We have described how this work extends existing works on single-robot anchoring. However, a number of works address similar problems and sub-problems. In particular, the *multi-target multi-sensor tracking* problem shares many similarities with the anchoring problem. Specifically, it requires solutions to a similar set of sub-problems – namely, data-association, information fusion, and prediction. The problem can roughly be stated as follows. Given a set of simultaneous measurements from M sensors, originating from an unknown number N of targets, estimate the state of the targets.

Data-association has mainly been explored in this context, and a vast literature has been devoted to the problem over many years – see [2] for a survey. Most of the work on data-association assumes that information consists of *sensor measurements*, usually of a commensurate type. And usually, only position information is considered in the algorithms. One can use existing data-association algorithms to address parts of the cooperative anchoring problem; however these methods should be extended to consider heterogeneous items of information, which may span many domains (e.g. position, color, shape, etc).

Information fusion is a wide area, and the field includes many methods which incorporate both perceptual and symbolic information [4], [11]. However, in the context of multi-target multi-sensor tracking, few approaches explicitly address the fusion of different types of information in complex domains. Also, in many works, the implications of having sensors with uncertain, non-fixed positions (as is the case with robotic systems) are not fully considered [5]. The cooperative anchoring problem thus requires the use of particularly flexible approaches to information fusion.

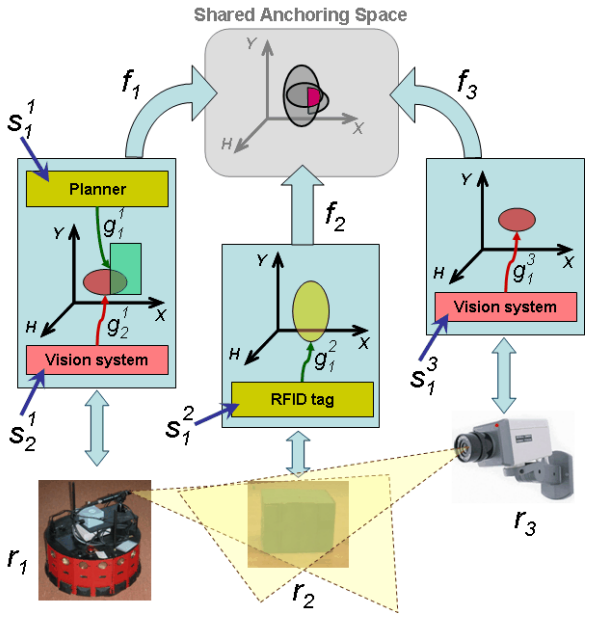

Fig. 1. Different elements of our framework in a parcel scenario similar to the example from the introduction.

III. A Computational Framework for Cooperative Anchoring

This framework extends existing single-robot anchoring frameworks [7], [6] in that it considers distributed systems and generic items of information. We will use the example scenario in Figure 1 to explain the framework.

A. Ingredients

The framework uses geometric spaces, called *anchor spaces*, to represent, compare and combine information. These are inspired by Peter Gärdenfors's *conceptual spaces* [9], which are well suited for dealing with both abstract, symbolic information and sensor-based, sub-symbolic information. An anchor space is a multi-dimensional space, which includes one dimension for each quality of interest (e.g. hue, x-position, etc). Independent groups of dimensions are grouped into *domains* (e.g. color, position, shape, etc).

Example 1: In Figure 1, there are two dimensions for the position domain (X, Y) plus one dimension for the color domain (H, for hue). Other dimensions could also be included (e.g., texture, shape, weight, etc); we restrict the example for graphical clarity. Also note that the position and color domains would normally be maintained separately.

For each robot r_i in the set $\{r_1, \ldots, r_M\}$, we define:

- A multi-dimensional *local anchor space* X_i, which is the space into which items of information are mapped; the space describes how local anchors are represented.
- A set $\{\alpha_1^i, \ldots, \alpha_{L_i}^i\}$ of local anchors: one for each object of which the robot is aware. Anchors are data structures, used to store the estimated values of the properties of objects. Each local anchor α_l^i contains the estimated properties of object o_l, considering only r_i's own information sources.
- A set $\{s_1^i, \ldots, s_{K_i}^i\}$ of *information sources*. Each s_j^i produces items of information in its domain Y_j^i.
- A set $\{g_1^i, \ldots, g_{K_i}^i\}$ of *grounding functions*, which map information from each source into the appropriate domains and dimensions in the local anchor space. That is, each g_j^i has a signature $g_j^i : Y_j^i \rightarrow X_i$.

In general, grounding functions map items of information into distribution clouds (e.g., probability distributions or fuzzy sets) in the anchor spaces, to account for uncertainty. For perceptual information, these functions can be seen as reverse sensor models. For symbolic information, they can act as predicate grounding relations [7]. In practice they are more general than either of these. We do not make assumptions about the origins of the functions: they can be hand coded, learned, or otherwise obtained. In robotics, sensor models and predicate grounding relations are normally given (or easily obtainable).

Example 2: In Figure 1, Pippi ($r1$) has two information sources: a symbolic task planner, and a vision system. The symbol system has a grounding function g_1^1 which maps symbols to regions in its local anchor space. For instance, the position symbol `entrance` is mapped to a rectangular area in the (X, Y) plane. The vision system has a grounding function g_1^2 that maps image regions to sets of position and color values. In the example, the estimated position and color of the box are mapped to an ellipsoid in the (X, Y, H) space. The shape of this set takes uncertainty into account. The RFID tagged parcel (r_2) and the ceiling camera (r_3) contain one information source each; these are mapped into their respective local anchor spaces.

We further define the following ingredients:

- A *global anchor space* X, a space into which information from each X_i can be mapped. The global space provides a common reference frame for the domains of interest, which is needed in order to match and fuse shared information. Typically, this space must be at least as rich as any individual X_i; otherwise information could be lost when mapping information to or from it.
- A set $\{\alpha_1, \ldots, \alpha_G\}$ of global anchors: one for each object of which any robot is aware. Each anchor α_g contains the estimated properties of object o_g.
- For each robot r_i, a function $f_i : X_i \rightarrow X$ which maps information from X_i to X. The functions f_i can involve cylindrical extensions and non-linear transformations, since the dimensionality and coordinate systems of X and X_i might differ.

Example 3: In Figure 1, all the anchor spaces (local and global) have the same dimensions. This means that the f_i transformations are the identity function; this is done in order to keep the example simple. Alternatively, the local spaces and the global space could be defined differently (e.g. some robots might use the RGB color space instead of HSV).

B. Anchor management

The cooperative anchoring problem can be seen as the problem of creating and maintaining a set of global anchors. In our framework, this problem is decomposed into two parts: the part involving the management of each individual robot's *local anchors*, and the part which involves the management of *global anchors*, shared across robots. Local anchor management can be done at a high frequency, since operations are done on-board the same robot. Global anchor management can depend on latencies in the communication network. This decomposition of the problem is natural given the distributed nature of the systems considered; it also helps reduce the computational complexity of the overall problem.

Both local and global anchor management are performed by sequentially addressing the two sub-problems mentioned earlier: data-association and fusion. Data-association relies on the metric structure of the anchor spaces to provide a means for information comparison: items which match in all domains can be considered to refer to the same object. Fusion involves combining these matched items, in all domains. As we noted earlier, both data-association and fusion are widely studied problems, and many solutions have been proposed. Our framework does not not require specific solutions for either. Later on we will discuss the approaches used here.

C. Local anchor management

1) Local data-association: Local data-association finds a partitioning of the set $\Phi^i(t)$, as discussed in section II-A, such that each partition $\Phi^i_l(t)$ contains all and only items of information referring to object o_l. Such a partition can contain at most one item from each information source, given our data-association assumptions. The items are also matched with all local anchors. If no anchor matches the items in a given partition, a new anchor is created; otherwise, the matching anchor is associated with the partition.

Example 4: In Figure 1, the items of information provided by the symbolic planner (s^1_1) and the vision system (s^2_1) overlap. Given the way information is represented in this example, we conclude that these items match, and therefore they could represent the same object.

2) Local information fusion: Local information fusion combines all items in $\Phi^i_l(t)$. If prediction were performed, the predicted state of anchor α^i_l could also be fused with the new information. The result of the fusion operation overwrites the contents of α^i_l.

Example 5: In Figure 1, the result of fusing the items from the previous example is represented by the intersection of the two areas; this yields a more accurate estimate of the object's position and/or color. This estimate is put in the local anchor α^1_1 for this object. The information in α^1_1 can then be used, for example, to control the motion of the robot.

It is important to realize that the index l in α^i_l is a local index, corresponding to the internal name used by robot r_i to denote object o_l, e.g., `object-14`. Other robots might use different internal names to refer to the same object. However there is an important exception. Some objects have *proper IDs*, which uniquely identify them. For instance, people have proper names, like 'Alex', and rooms can be labeled, like 'T-234'. Some items of information may bear the proper ID of the object to which they refer; e.g., `(age Alex 22)`. We make sure that all items of information bearing the same proper ID are combined into the same anchor. This is done by modifying the data-association step so that it forces items with the same proper ID into the same partition. If any items put into an anchor have a proper ID, this ID is stored in the local anchor, so that it is also considered during global anchor management.

D. Global anchor management

1) Global data-association: Global data-association finds a partitioning of the set of all local anchors, $A = \{\alpha^i_l \mid i = 1, \ldots, M, l = 1, \ldots, L_i\}$, such that each partition A_g contains at most one anchor from each robot (since no two anchors in the same robot refer to the same object), and in such a way that local anchors in the same partition match in all domains. Intuitively, each partition should include all the local anchors that represent the same object o_g. Some local anchors may have proper IDs; again, we use these IDs to constrain the data-association step, so that local anchors with the same IDs are forced into the same partition.

2) Global information fusion: Once the correspondence among local anchors has been established, we combine the information contained in the local anchors which refer to the same objects. This yields a global estimate of the properties of each object o_g. The result of the fusion of all anchors in A_g is stored in the corresponding global anchor α_g. Previous global anchors are not fused with the local anchors; state information is considered only during local fusion. Note that these global anchors can be maintained locally in each robot, or in a centralized manner. More will be said about this later.

Example 6: In Figure 1, Pippi (r_1), the parcel (r_2) and the ceiling camera (r_3) have each created local anchors for the box. These are respectively called α^1_1, α^2_{14} and α^3_8. Their information contents are represented by the three gray areas in the global anchor space. Since these areas overlap, all three anchors are put into the same partition of A, say A_6. The combination of these is represented by their intersection, and this is stored in the global anchor α_6, which represents the global estimate of the box's properties.

Intuitively, the index g of the α_g anchor corresponds to a *global name* which can be used by the various robots to refer to object o_g. If any of the anchors in A_g has an associated proper ID, this ID is copied into the global anchor α_g: this associates the global name of the anchor in the system to the proper name of the object in the world. In practice, reliably naming global anchors can be difficult if unreliable communication channels are used, but there are distributed algorithms which address this [10].

IV. Implementation

We have implemented the framework described previously using techniques from fuzzy logic [12]; this is motivated by a number of factors. Fuzzy sets are able to represent many types of information and uncertainty (e.g. imprecision, vagueness, ambiguity, unreliability, etc). Also, fuzzy logic provides a formal mechanism for matching and fusing information, and it includes a rich set of operators for both of these [3]. Fuzzy logic also lends itself to relatively straightforward and computationally efficient implementations.

Information in the anchor spaces is represented by n-dimensional fuzzy sets. A fuzzy set F over the n-dimensional space S is characterized by a membership function

$$\mu_F : S \rightarrow [0, 1]$$

which gives the degree of membership $\mu_F(x)$ of each point x in S to the fuzzy set F. So a grounding function g_j^i maps information from the information source s_j^i to a fuzzy set over the local anchor space X_i.

Each fuzzy set F is implemented as an n-dimensional array of floats, where the value in each element x corresponds to the value of $\mu_F(x)$. For example, color information can be represented using a 3D space, with dimensions for hue, saturation, and lightness, while 2D position information can be represented using a grid of (x, y) values. If necessary, some dimensions can be approximated to reduce complexity.

In the implementation used in this work, the dimensionality and units of the global space X are the same as those used in the local spaces X_i. This implies that for each robot r_i, the mapping function f_i is the identity function. This choice does not limit the generality of the framework; it merely simplifies the implementation, by affecting how and when certain computations are made.

The global anchor space could be implemented in a centralized way: individual robots could send their local anchors to this central location, which should then take care of matching and fusing these, as described earlier. The results would then be sent back to the individuals. This approach would strongly rely on the communication framework. Instead, our implementation distributes the global anchor space among the robots. The robots send and receive anchors to each other directly, in a peer-to-peer fashion [15]. Locally, each robot matches and fuses local anchors received from other robots, and creates and maintains global anchors accordingly. This approach is more robust than a centralized one, since any successfully communicated information can be exploited.

Matching of information is implemented as follows. If μ_1 and μ_2 are two fuzzy sets on the same space X, representing two distinct items of information, then the degree of matching between these two items is given by:

$$\text{match}(\mu_1, \mu_2) \;=\; \sup_{s \in S} (\mu_1(s) \otimes \mu_2(s))$$

where sup denotes the supremum (least upper bound) operator, and $\otimes$ is a triangular norm, or T-norm [12]. The most common T-norms used in fuzzy logic are the minimum and product operators. In the matching done in the experiments reported below we use the minimum operator, since its result is independent of the number of items being matched. Match values are computed for each domain separately, and the minimum of these is used as the overall match value between items. Figure 2 shows an example of two 1-dimensional fuzzy sets. The result of matching these two fuzzy sets as described is shown by the dotted line.

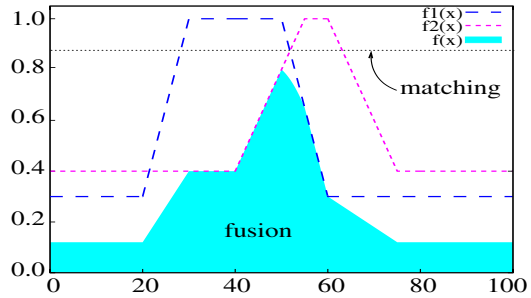

Fig. 2. The result of fusing f1(x) and f2(x) using the product T-norm is shown by the filled fuzzy set f(x). The degree of matching of the two fuzzy sets is shown by the dotted line.

Data association is performed by enumerating all possible associations and computing the value of each overall association hypothesis as the average of the degrees of matching between the included associations. This average is biased since we discard associations with match values below a certain threshold. We also disambiguate associations with the same match value by preferring associations with many items – this encourages robots to agree about objects whenever possible. The hypothesis with the highest value is used; this is essentially a single-frame *Global Nearest Neighbour (GNN)* approach [2].

Fusion is implemented using fuzzy intersection. If μ_1 and μ_2 are fuzzy sets, the result of fusing these items is represented by the fuzzy set μ given, for any $x \in X$, by

$$\mu(x) = \mu_1(x) \otimes \mu_2(x).$$

In the experiments reported below, we use the product T-norm for fusion operations, since it emphasizes values which are common to all sources. In Figure 2, the result of the fusion as described is the filled in area. More details about the fusion algorithm can be found in [5].

Other operators could be used for both matching and fusion, and several alternatives can be found in the literature [12], [3]. The choice typically depends on the types of information and information sources being considered.

V. An Illustrative Experiment

The purpose of this experiment is to show how different types of information arriving at different times from different robots are anchored using our framework. The scenario is inspired by the example in the introduction.

For many of our experiments we use a small apartment (about $25\,\mathrm{m}^2$) as a testbed, in which various intelligent devices are embedded (e.g. RFID readers, localization systems, robots, etc). The experiment presented here is carried out in a gazebo [1] simulation of this apartment, shown in Figure 3. In both real and simulated experiments, the devices in the

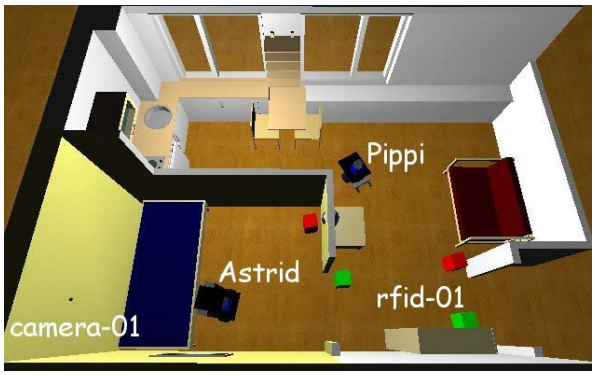

Fig. 3. The gazebo simulation of the apartment used in the experiment.

environment are interconnected using middleware developed by us as part of our PEIS-Ecology project [15].

The participants in the experiment are the following: r_1 is an RFID reader called *rfid-01*; r_2 is a mobile robot named *Astrid*; r_3 is a mobile robot named *Pippi*; r_4 is a fixed ceiling camera called *camera-01*. All the participants are considered to be robots, or at least subsets of robotic systems. *Pippi* uses two information sources in this scenario: a symbolic task planner, and a vision system. *Astrid* and *camera-01* use only their vision systems, and the RFID reader provides symbolic information obtained from detected RFID tags. The RFID reader is placed near the entrance of the apartment, and it can detect RFID tags within about one meter from the door. The objects being observed are two red and two green parcels. One of the green parcels has an RFID tag with ID *parcel-21*. Note that the number and properties of these parcels are not known to the participants.

The three domains used in this experiment are: a (discrete) 1D texture domain, a 2D position domain, and a 3D color domain. The coordinate systems used are shown at the top of Figure 4. Texture information has been added to the data artificially, since the vision systems we use are not able to detect textures. Recall that the local and global anchor spaces have the same dimensions and units. In practice this means that position information obtained in local coordinates (e.g. from vision) is converted to global coordinates before being put into the anchor spaces. The scenario begins with *Pippi* being told to fetch *parcel-21* which is near the entrance; events unfold as follows (see Figure 4).

Information 1 (Symbolic): Rfid-01 provides a local anchor α_1^1 with symbolic texture and position information, as shown at time *t1* in Figure 4. This anchor contains the information that the parcel is striped, and in the region covered by the RFID reader. Since *Pippi* knows the ID of the parcel (the ID is a *proper ID*), it can create a global anchor for *parcel-21*, which for now is the same as α_1^1.

Information 2 (Perceptual): Astrid sees three parcels, and provides three corresponding local anchors, as shown at *t2* in Figure 4. The anchor α_1^2 does not match *parcel-21*'s texture. The anchor α_3^2 does not match *parcel-21*'s position. Anchor α_2^2 matches both in the texture and position domains, and is fused with the previous information. The global anchor now also contains color information about the parcel, based on the color of α_2^2. However, the position estimate for *parcel-21* has not improved, since *Astrid* was poorly localized; this can be seen from the fact that all three local anchors have very imprecise position information.

Information 3 (Perceptual): Pippi detects two parcels, as shown at *t3* in Figure 4. Both local anchors match the position estimate for *parcel-21*; however, only α_1^3 matches the color information provided by α_2^2. This anchor is therefore combined with the previous information, and the global anchor now has an improved estimate of *parcel-21*'s position.

Information 4 (Perceptual): Camera-01 sees four parcels. Only α_1^4 matches the position estimate for *parcel-21*, and only this anchor is shown at time *t4* in Figure 4. This information is combined with the previous information, yielding a very accurate position estimate for *parcel-21*.

VI. CONCLUSIONS

The main contribution of this paper is the definition of both the cooperative anchoring problem, and a general computational framework to address it. The problem shares some common points with multi-target multi-sensor tracking and other similar problems; however, the problem is different, mainly because of its emphasis on the principled integration of multiple, highly heterogeneous sources of information.

The experiment shown in this paper is meant to illustrate, not validate, our framework. Future work will involve a systematic validation of both the generality of the framework, and its effectiveness in reducing uncertainty and ambiguity. We plan to use a progression of scenarios, using both real robots and simulations, to do this.

REFERENCES

[1] Player/Stage/Gazebo website. http://player.sourceforge.net.
[2] S.S. Blackman. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Mag*, 19(1):5–18, 2004.
[3] I. Bloch. Information combination operators for data fusion: A comparative review with classification. *IEEE Trans on System, Man and Cybernetics*, A-26(1):52–67, 1996.
[4] I. Bloch and A. Hunter. Fusion: General concepts and characteristics. *Int Journal of Intelligent Systems*, 16(10):1107–1134, 2001.
[5] J-P. Cánovas, K. LeBlanc, and A. Saffiotti. Robust multi-robot object localization using fuzzy logic. In D. Nardi, M. Riedmiller, and C. Sammut, editors, *RoboCup 2004*. Springer-Verlag, 2004.
[6] A. Chella, S. Coradeschi, M. Frixione, and A. Saffiotti. Perceptual anchoring via conceptual spaces. In *Proc. of the AAAI-04 Workshop on Anchoring Symbols to Sensor Data*, San Jose, CA, 2004.
[7] S. Coradeschi and A. Saffiotti. Anchoring symbols to sensor data: preliminary report. In *Proc. of the 17th AAAI Conf.*, pages 129–135, Austin, TX, 2000.
[8] S. Coradeschi and A. Saffiotti. An introduction to the anchoring problem. *Robotics and Autonomous Systems*, 43(2-3):85–96, 2003.
[9] P. Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. MIT Press, Cambridge, MA, USA, 2000.
[10] S. Ghosh. *Distributed Systems: An Algorithmic Approach*. Chapman And Hall/CRC, Boca Raton, FL, USA, 2007.
[11] DL Hall and J. Llinas. An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23, 1997.
[12] G. J. Klir and T. A. Folger. *Fuzzy sets, uncertainty, and information*. Prentice-Hall, 1988.
[13] J.H. Lee and H. Hashimoto. Intelligent space – concept and contents. *Advanced Robotics*, 16(3):265–280, 2002.
[14] Network Robot Forum. www.scat.or.jp/nrf/English/.
[15] A. Saffiotti and M. Broxvall. PEIS ecologies: Ambient intelligence meets autonomous robotics. In *Proc of the Int Conf on Smart Objects and Ambient Intelligence (sOc-EUSAI)*, pages 275–280, Grenoble, France, 2005.

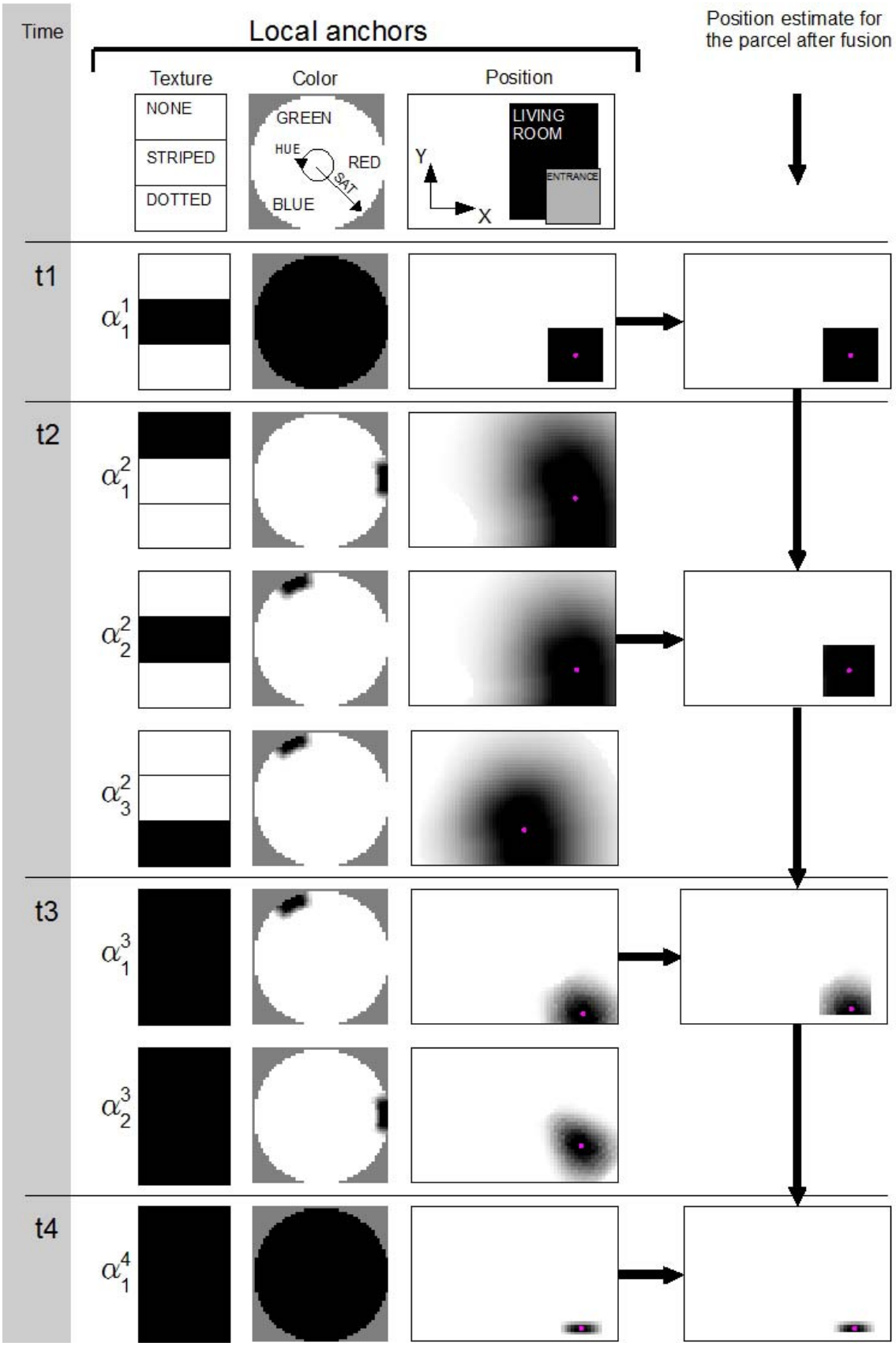

Fig. 4. The contents of local anchors exchanged at given times are shown in the three columns labelled *texture*, *color*, and *position*. The coordinate systems used for these domains are shown at the top of the figure. Possible values are drawn dark, impossible values are light. At time t1, one local anchor is provided by *rfid-01* (r_1). At time t2, three local anchors are provided by *Astrid* (r_2). At time t3, two local anchors are provided by *Pippi* (r_3). At time t4, *camera-01* (r_4) provides four local anchors, only one of which is shown. At each time, the anchor which matches the current information about *parcel-21* is fused with previous information, and the resulting global anchor's position information is shown on the right.