Modelling Language, Action, and Perception in Type Theory with Records

Simon Dobnik, Robin Cooper, and Staffan Larsson

Department of Philosophy, Linguistics and Theory of Science
University of Gothenburg, Box 200, 405 30 Göteborg, Sweden
simon.dobnik@gu.se, {staffan.larsson,robin.cooper}@ling.gu.se
http://www.flov.gu.se

Abstract. Formal models of natural language semantics using TTR (Type Theory with Records) attempt to relate natural language to perception, modelled as classification of objects and events by types which are available as resources to an agent. We argue that this is better suited for representing the meaning of spatial descriptions in the context of agent modelling than traditional formal semantic models which do not relate spatial concepts to perceptual apparatus. Spatial descriptions include perceptual, conceptual and discourse knowledge which we represent all in a single framework. Being a general framework for modelling both linguistic and non-linguistic cognition, TTR is more suitable for the modelling of situated conversational agents in robotics and virtual environments where interoperability between language, action and perception is required. The perceptual systems gain access to abstract conceptual meaning representations of language while the latter can be justified in action and perception.

Keywords: language, action, perception, formal semantics, spatial descriptions, learning and classification.

1 Introduction

Classical model-theoretic semantics [1] does not deal with the connection between linguistic meaning and perception or action. It does not provide a theory of non-linguistic perception and it is not obvious how to extend it to take account of this. Another feature of classical approaches to model theory is that they treat natural language as being essentially similar to formal languages in, among other things, assuming that there is a fixed interpretation function which mediates between natural language and the world. But how this function is determined or modified dynamically by experience is left unaccounted for.

Furthermore, as has been treated in classical theories, there are many cases where the meaning of linguistic expressions is underspecified and can only be recovered from the context. Deictic pronouns are a well known example – they must be resolved against a contextually present entity which can typically be recovered from the discourse. Spatial descriptions such as 'to the left of' or

D. Duchier and Y. Parmentier (Eds.): CSLP 2012, LNCS 8114, pp. 70–91, 2013.

'near' rely on the physical context to an even greater extent as they must be evaluated against a particular pair of objects, their geometric representation and arrangement and the size of the scene. At the same time they also rely on our world knowledge about the interaction of objects.

Previously, there has not been a unified model of spatial language. Proposals include first order logic representations that we find in [2] and [3]. These only represent the conceptual knowledge but ignore how this knowledge maps to the environment. On the other hand there are approaches such as [4] and [5] that completely ignore the conceptual dimension of spatial descriptions and only consider perceptual parameters such as distances and angles in two or three dimensional space.

A need for an integrated model of language, action and perception is expressed most strongly in robotics ([6], [7], [8] and [9]). They are concerned to represent and apply linguistic knowledge in robotic control (of a particular system) but they are less concerned with developing a general theory of meaning. [10] describes a system where mobile robots situated in a real room learn the meaning of perceptual descriptions as classifiers which they use to generate descriptions of new scenes or to answer questions about the scene and perform actions. In this paper we build on this experience by attempting to re-formulate it in a theoretical framework known as TTR (Type Theory with Records) [11,12] which can represent perceptual, conceptual and discourse constraints that make up the meaning of words of natural language. This way we take an important step towards a more cognitive model of robotic perception and control.

2 TTR: A Theory of Perception and Natural Language Semantics

Types are intensional – that is, there can be distinct types which have identical extensions. They are semantic categories some of which can be linked to perception. Perception involves the assignment of a type to an object or event, a *judgement* that an object a is of type T (in symbols, $a : T$), that is, in the case of a perception, a is perceived as an object of type T. The notion of truth is linked to such judgements. In particular types can be used to model propositions. A type T is 'true' just in case there is something a such that $a : T$. An object a of type T is also known as a *proof object* of T or a *witness* for T.

The theory defines a few basic types such as *Ind* and *Ev* which correspond to basic human conceptual categories such as individuals and events. But there are also complex types built from simple types and from other complex types. Most important among these are *record types* which are similar to feature structures which contain label-value pairs. Proof objects of record types are called records. Here is an example of what a record type *Apple* might be:

$$\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_{\mathrm{apple}} & : & \mathrm{apple(x)} \\ \mathrm{c}_{\mathrm{ash}} & : & \mathrm{apple\text{-}shape(x)} \\ \mathrm{c}_{\mathrm{ac}} & : & \mathrm{apple\text{-}colour(x)} \\ \mathrm{c}_{\mathrm{at}} & : & \mathrm{apple\text{-}taste(x)} \\ \mathrm{c}_{\mathrm{as}} & : & \mathrm{apple\text{-}smell(x)} \end{bmatrix}$$

The letter 'x' in labels introduces an identifier and 'c' in labels stands for constraints. The example introduces types which are constructed from predicates (like 'apple') and objects which are arguments to these predicates. An object belonging to such *p-type* can be considered to be an event or a situation [13,14]. It follows that types can be used as propositions which are true just in case there is an object of that type. Notations like 'apple(x)' in the above example are actually convenient abbreviations for a pair of objects:

$$\langle \lambda v{:}\mathit{Ind}(\mathrm{apple}(v)),\ \langle \mathrm{x} \rangle \rangle$$

which consists of a dependent type, an n-ary function which maps objects to a type, and a sequence of paths indicating where the arguments to this function are to be found. The idea is that when you are checking whether a record is of this type, you apply the dependent type to the values you find at the end of the indicated paths in the record which is being checked. Thus if we are checking that the record

$$\begin{bmatrix} \mathrm{x} & = & a \\ \mathrm{c}_{\mathrm{apple}} & = & p_1 \\ \mathrm{c}_{\mathrm{ash}} & = & p_2 \\ \mathrm{c}_{\mathrm{ac}} & = & p_3 \\ \mathrm{c}_{\mathrm{at}} & = & p_4 \\ \mathrm{c}_{\mathrm{as}} & = & p_5 \end{bmatrix}$$

is of the above record type then we need to check that $a : \mathit{Ind}$, $p_1 : \mathrm{apple}(a)$ and so on.

As demonstrated by this example the type representation combines both conceptual ($\mathrm{c}_{\mathrm{apple}}$) and perceptual constraints ($\mathrm{c}_{\mathrm{ash}}$) that must be verified against records containing individuals in order that these records may be judged as of type *Apple*. The model brings the semantic theory for natural language and models of perception and action in robotics close together. The important consequence of such a combined model is that natural language semantics does not only involve conceptual or categorial knowledge but also distributional knowledge. Linguistically, this model has, among other things, advantages in the treatment of intensionality, vagueness and dialogue structure [15,16,17,18]. [19] applies TTR to model how perceptual categories are learned through dialogue interaction and observations. The TTR model also contributes to artificial perception as used for example in robotics since this is directly related to conceptual knowledge represented in language which recently has been argued to be

necessary for a more successful image recognition [20]. Overall, TTR brings formal semantics closer to fundamental human cognition than has been attempted in classical model-theoretic approaches.

The mapping between the perceptual and conceptual domain can only be learned through experience as a probabilistic classifier – it is unclear otherwise how one would specify a mapping from a continuous numeric domain governed by distributions to a symbolic domain with a sufficient level of complexity required to model human spatial cognition. An agent does not have direct access to types in another agent's mental structure but can only observe them through interaction with the other agent. Furthermore, some concepts are inherently vague. As demonstrated later with spatial relations humans seem to tolerate divergences when making this mapping where some parts of the continuum are more likely to be mapped to a particular category than others but this also means that some parts may be mapped to more than one category simultaneously. One can speak of mapping tendencies rather than categorical rules and such tendencies can only be learned through observations as probability distributions. Finally, as we will demonstrate in our discussion of robotic mapping and localisation the vagueness in mapping may be introduced through the noise of sensors and actuators.

3 Spatial Descriptions

By spatial descriptions we mean two or three place prepositional predicates that refer to the location of objects usually known as the *located object* and the *reference object* [21] or *referent* and *relatum* [3] or *figure* and *ground* [22]. Examples are: on(spider,wall) and between(apple,vase,glass). Events may also be arguments of such predicates as shown by in(eat(George,apple),garden) or in(put(book),bag). The preceding references provide a survey of semantics of spatial descriptions in English. Roughly, its constraints fall under four headings discussed below.

3.1 Scene Geometry

Scene geometry involves a two-dimensional or three-dimensional coordinate frame in which we can represent objects and angles and distances between them, as modelled for example in the *attentional vector sum model* [23]. Although we have strict geometric notions of 'left' and 'right' that coincide with the axes of the coordinate frame we also seem to tolerate substantial divergences from these geometric meanings. Thus, within a coordinate frame we can identify areas of different degrees of applicability of a spatial description. It necessarily follows that for a certain area more than one description will be applicable but most likely to a different degree. In this respect, spatial descriptions may be considered vague. The area of applicability is typically modelled with spatial templates [5]. These map an invisible grid on the spatial continuum. The reference object is always in the centre of the grid and the located object is placed in one of the cells. Human judges then evaluate the applicability of the spatial description given a particular placement of the located object. Over several judgements

different cells come to have different mean applicability ratings thus capturing distributional knowledge within a cell and between the cells.

A spatial template holding between the reference object and the located object may be influenced by other objects in the scene known as distractors. Imagine a situation where B is behind and slightly to the left of A and another situation where A and B are located the same as before but there is another object C which is directly behind A. While in the first case one would describe the scene 'B is behind A'. In the second such description would become unacceptable since C better fulfils the behind relation.

3.2 Objects as Geometric Conceptualisations

Spatial descriptions do not refer to the actual objects in space but to their geometric representations which may be points, lines, areas and volumes as shown by the following examples adapted from [21]. A description 'the crack in the vase' refers to a situation where the vase is conceptualised as a surface whereas a description 'the water in the vase' refers to the vase as a volume enclosed by the object. Different conceptualisations arise from the way objects interact with each other and thus depend on our world knowledge but they may also reflect our view on the situation as for example 'under the water (surface)' and 'in the water (volume)'.

3.3 World Knowledge

The influence of world knowledge may go beyond conceptualisation as shown by the following examples from [21]. In #'The Empire State building is near Mary' the problem is selecting an appropriate reference object - one that is salient enough and static so that it can determine the location of the first object. In cases like 'The flowers are in the vase', 'The child is in the back of the car' and 'The queue is at the counter' the geometrical grounding criteria for 'in the vase' and 'in the back of the car' appear to be falsified – the flowers are not entirely contained in the volume of the vase and the child is in the back of the seating area of the car and not in the booth – and yet the descriptions are acceptable in the intended situations.

Furthermore, the contribution of the geometrical and world knowledge component to the meaning of spatial descriptions in not always equal. For example, [24] show in the experiments with human observers of images of a man in the rain holding an umbrella where the umbrella is providing a varying degree of protection from the rain that 'above' is more sensitive to the geometrical component than 'over' and 'over' is more sensitive to the object function component than 'above'. Descriptions of 'the umbrella is over a man' was considered acceptable even in cases where the umbrella was held horizontally but was providing protection from the rain.

3.4 Perspective or Reference Frame

Spatial relations can be subdivided into two groups: (i) topological spatial relations such as *near* where geometrically only a distance between the two objects is taken into consideration and (ii) directionals such as *to the left of* which also require a model of perspective or the reference frame. Given a fixed scene containing a chair and a table, the table may be described to be 'to the left of the chair', 'to the right of the chair', 'behind the chair' or 'south of the chair'. The perspective is determined by some point in the scene called the viewpoint. There are three ways in which the viewpoint is set in human languages [25]: (i) relative reference frame: by some third object distinct from the located and reference objects (the speaker, the hearer, the sofa); (ii) intrinsic reference frame: by the reference object (the chair); or (iii) extrinsic reference frame: by some global reference point (the North). The viewpoint contains three geometric components: (i) a set of orthogonal axes, (ii) a point of origin, and (iii) an orientation vector [26]. The geometric spatial templates or potential fields are projected within the framework defined by the viewpoint. The viewpoint may be described linguistically 'from your view' or 'from there' but it is frequently left out. This can be cases where it can be inferred from the perceptual context, for example, if given some configuration of the scene a spatial description cannot be ambiguous. It follows that when interpreting and generating spatial descriptions humans must verify spatial templates according to different viewpoints which requires considerable computational complexity [27]. Secondly, the viewpoint parameter may be negotiated and aligned between conversational partners who use it over several utterances in conversation as shown experimentally in [28]. The viewpoint is thus a discourse parameter which can also modelled in TTR and is intended for our future work (for negotiation of conceptual knowledge in discourse see for example [17]).

4 A TTR Account of Spatial Descriptions

In this section we discuss how the constraints described in Sections 3.1 to 3.4 can be expressed in TTR to represent meaning for a robotic agent.

4.1 Maps of the Environment and Geometric Object Representations

In robotics the location of the robot and other objects in its environment is frequently determined using a method known as SLAM which stands for 'simultaneous localisation and map building' [29]. A robot starts in an unknown location and in an unknown environment and uses its relative observations of distances to landmarks to build a global absolute map of the environment incrementally using probabilistic unsupervised machine learning. The robot is mobile which means that its location changes in time but the landmarks are static. Therefore, the global map is modelled as a sequence of temporarily dependant

states. When the robot transitions from a state t to a new state at $t+1$, the new state is modelled as being dependent on the previous state at t and some process model. In each state the robot makes relative observations of landmarks with some observation model which are related to the location states.[1] In TTR both robot states and actions can be modelled as belonging to the types *RobotState* and *Action* respectively and update functions as being of the following type: *RobotState* $\rightarrow$ (*Action* $\rightarrow$ *RecType*). The idea of this is that given a robot state and an action the update function will return the type of the updated robot state. We define the type *RobotState* to be:

$$\mathit{RobotState} = \begin{bmatrix} \text{a} & : & \mathit{Ind} \\ \text{c}_{\text{robot}} & : & \text{robot(a)} \\ \text{pm} & : & \mathit{PointMap} \\ \text{c}_{\text{loc}} & : & \text{located(a,pm[0])} \\ \text{objects} & : & [\mathit{Object}(\text{pm})] \\ \text{c}_{\text{object\_map}} & : & \text{object\_map(objects, pm)} \\ \text{beliefs} & : & \mathit{RecType} \\ \text{heading} & : & \mathit{Real} \\ \text{time} & : & \mathit{Time} \end{bmatrix}$$

Here the field labelled 'a' is for the robot itself as required by the constraint labelled 'c_{robot}'. The type *Ind* 'individual' is a basic type. However, the robot has to figure out for itself what counts as an individual. Part of the novelty of our account here is that we try and treat this individuation process using the same tools as we use for semantics. The identification of individuals is normally assumed in classical approaches to semantics.

The 'pm'-field is for a point map, that is, a vector of points. This represents the points where the robot has discovered that there is something present, for example, as a result of a laser emitted by a sensor being reflected. A laser scanner may receive reflections from many objects but only those that are static enough and distinct enough over several scans are good to be included as landmarks on the map. We will model vectors in our type theory as lists. Thus the type *PointMap* is defined by

$$\mathit{PointMap} = [\mathit{Point}]$$

That is, an object of the type *PointMap* is a list of points, that is a list all of whose members are of the type *Point*. Points are defined as a pair (for 2D) or a triple (for 3D) of coordinates (real numbers). We code these tuples in TTR as records with fields labelled 'x, 'y' and 'z' following the standard names of coordinates:

$$\mathit{Point} = \begin{bmatrix} \text{x} & : & \mathit{Real} \\ \text{y} & : & \mathit{Real} \end{bmatrix} \text{(for 2D)}$$

[1] Both models include parameters such as controls used to perform actions and noise.

$$\mathit{Point} = \begin{bmatrix} \mathrm{x} & : & \mathit{Real} \\ \mathrm{y} & : & \mathit{Real} \\ \mathrm{z} & : & \mathit{Real} \end{bmatrix} \text{ (for 3D)}$$

The first element of the point map is defined to be the location of the robot, as required by the constraint labelled '$\mathrm{c}_{\mathrm{loc}}$'. (If L is a list, we use the notation $L[n]$ to represent the nth member of L.)

The 'objects'-field is for an object map based on the point map. This is a list of objects that the robot has hypothesised on the basis of the points it has observed and recorded in the point map. An object is either a point object, an object located at a point, or a region object, an object located at a region, or a volume object, an object located in a volume (object conceptualisations in Section 3.2). These can be different views of the same actual object where we regard its location as either a point or a region of points or a volume of points. The types *Object*, *PointObject* and *RegionObject* are dependent types: they are functions from point maps to types. We define the type *Object* to be a join type (a disjunctive type):

$$\mathit{Object} = \lambda p{:}\mathit{PointMap}(\mathit{PointObject}(p) \wedge \mathit{RegionObject}(p))$$

The dependent types *PointObject* and *RegionObject* are defined as follows:

$$\mathit{PointObject} = \lambda p{:}\mathit{PointMap}(\begin{bmatrix} \mathrm{a} & : & \mathit{Ind} \\ \mathrm{pnt} & : & \mathit{Point} \\ \mathrm{c}_{\mathrm{point}} & : & \mathrm{observed\_point}(\mathrm{pnt},p) \\ \mathrm{c}_{\mathrm{loc}} & : & \mathrm{located}(\mathrm{a},\mathrm{pnt}) \end{bmatrix})$$

$$\mathit{RegionObject} = \lambda p{:}\mathit{PointMap}(\begin{bmatrix} \mathrm{a} & : & \mathit{Ind} \\ \mathrm{reg} & : & \mathit{PointMap} \\ \mathrm{c}_{\mathrm{region}} & : & \mathrm{region}(\mathrm{reg},p) \\ \mathrm{c}_{\mathrm{loc}} & : & \mathrm{located}(\mathrm{a},\mathrm{reg}) \end{bmatrix})$$

π is an observed point in a point map, p, just in case for some $n > 0$, $\pi = p[n]$:

$$r{:}\mathrm{observed\_point}(\pi,p) \text{ iff } r{:} \begin{bmatrix} \mathrm{n} & : & \mathit{Nat} \\ \mathrm{c}_{=} & : & \pi = p[\mathrm{n}] \end{bmatrix}$$

A region in a point map is a subset of the points in the point map which meet certain conditions. We code the conditions in TTR as follows:

$$r{:}\mathrm{region}(p_1,p_2) \text{ iff } r{:} \begin{bmatrix} \mathrm{n} & : & \mathit{Nat} \\ \mathrm{c}_{\mathrm{length}} & : & \mathrm{length}(p_1,\mathrm{n}) \\ \mathrm{c}_{\geq 2} & : & \mathrm{n} \geq 2 \\ \mathrm{c}_{\mathrm{plane}} & : & \mathrm{on\_plane}(p_1) \\ \mathrm{c}_{\mathrm{hull}} & : & \mathrm{convex\_submap}(p_1,p_2) \end{bmatrix}$$

A point map is on a plane just in case all the points in the point map have the same ℓ-coordinate where ℓ is either 'x', 'y' or 'z'. A point map, p_1, is a convex submap of a point map p_2 just in case the points in p_1 are a subset of points in p_2 and the points in p_1 can be enclosed with a convex hull. Not every possible convex submap ≥ 2 represents a useful region object. Judging what is an object in the world is complex and depends on its physical properties, its function and its interaction with other objects and the world as observed by humans. Thus, we want our types for objects approximate human types since other organisms (e.g. a bee) may observe the same situations differently. On a point map different macro objects may appear as different densities and arrangements of point objects since they would typically have different light reflection properties. Sometimes the points are enriched by colour from camera images fused with point clouds which adds another another discriminating constraint for selecting convex hulls.[2].

The $c_{\text{object\_map}}$-field is for constraints that we may wish to place on the object map as a whole. Thus, for example, if for some point map p, if o_1 : *PointObject*(p), o_2 : *RegionObject*(p) and o_1.pnt is a member of o_2.reg, then we will want to require that o_1.a=o_2.a. This means that the actual hypothesised individual in these cases is the same. These are a point view and a region view of the same object. In terms of TTR:

$$r{:}\text{object\_map}(\mathit{objs},\mathit{pm}) \text{ implies } r{:}\neg \begin{bmatrix} \mathrm{n}_1 & : & \mathit{Nat} \\ \mathrm{n}_2 & : & \mathit{Nat} \\ \mathrm{o}_1 & : & \mathit{PointObject} \\ \mathrm{o}_2 & : & \mathit{RegionObject} \\ \mathrm{c}_{\mathrm{o}_1\mathrm{n}_1} & : & \mathrm{o}_1{=}\mathit{objs}[\mathrm{n}_1] \\ \mathrm{c}_{\mathrm{o}_2\mathrm{n}_2} & : & \mathrm{o}_2{=}\mathit{objs}[\mathrm{n}_2] \\ \mathrm{n}_3 & : & \mathit{Nat} \\ \mathrm{c}_{\mathrm{o}_1\mathrm{o}_2} & : & \mathrm{o}_1.\mathrm{pnt} = \mathrm{o}_2.\mathrm{reg}[\mathrm{n}_3] \\ \mathrm{c}_{\neq} & : & \mathrm{o}_1.\mathrm{a} \neq \mathrm{o}_2.\mathrm{a} \end{bmatrix}$$

Semantically, this step corresponds to the first stage of the *object reification* as the point landmarks are primitive geometric representations of objects. One could object to our treatment of points as being primitives of geometric representations of individuals. However, linguistically we do refer to whole objects as if they were points in which case we normally take the point that represents their centre of mass, for example 'This point on the map is the museum' or 'The town is at the border' (conceptualised as a point at a line). Objects also display the effect of granularity as we may think of complex objects as consisting of simple ones which are their sub-parts. For example, accumulating grains of wheat makes a heap although the exact boundary after how many grains this happens is vague as exemplified by the well-known sorites paradox. For the SLAM procedure itself such point landmarks are sufficient representations of objects since

[2] See for example the resources and tools available at The Point Cloud Library (PCL) at `www.pointclouds.org`

it is only them that are tracked between different states. However, for humans cognition more complex representations are required. As just stated, even if an object is thought of as a point we do not take just any point. On the other hand, the point landmark representations are very accurate geometric representations of spatial topology, much more than representations from camera images for example, and therefore highly suitable for representing spatial relations as demonstrated in [10].

There is a another constraint related to the individuation process, namely the objects must be discrete. The constraint c_{object_map} also ensures that there are no points on the point map that represent two objects.

$$r{:}\text{object\_map}(objs,pm) \text{ implies } r{:}\neg \begin{bmatrix} n_1 & : & Nat \\ n_2 & : & Nat \\ o_1 & : & RegionObject \\ o_2 & : & RegionObject \\ c_{o_1n_1} & : & o_1{=}objs[n_1] \\ c_{o_2n_2} & : & o_2{=}objs[n_2] \\ c_{\neq} & : & o_1.a \neq o_2.a \\ c_{overlap} & : & \text{overlap}(o_1, o_1, pm) \end{bmatrix}$$

The 'beliefs'-field is for a record type which will represent the robot's 'beliefs' about the objects in its object map.

The 'heading'-field is for the number of degrees which the robot deviates from its initial heading (i.e. the direction in which it is pointing), represented as a real number. The 'time'-field is for a natural number representing the number of observations the robot has made since the beginning of the current run. Thus the type *Time* can here be construed as the type of natural numbers.

The type of the initial state of the robot can be represented as below.

$$InitRobotState = \begin{bmatrix} a & : & Ind \\ c_{robot} & : & \text{robot(a)} \\ pm = [\begin{bmatrix} x = 0 \\ y = 0 \end{bmatrix}] & : & PointMap \\ c_{loc} & : & \text{located(a,pm[0])} \\ \text{objects=[]} & : & [Object(\text{pm})] \\ c_{object\_map} & : & \text{object\_map(objects, pm)} \\ \text{beliefs=[]} & : & RecType \\ \text{heading=0} & : & Real \\ \text{time=0} & : & Time \end{bmatrix}$$

Note that the labels 'pm', 'objects', 'beliefs', 'heading' and 'time' have particular values assigned which we model with manifest fields. The robot has not observed any landmarks yet and hence its point map will only contain one element, the

location of itself expressed as the origin of a two-dimensional coordinate space. All subsequent locations of landmarks will be made relative to this origin.

Actions involve an application of sensors and actuators from which we can continuously read values. An action is also performed at a particular time. Since it is performed by the agent itself rather than observed, the controls that were used to perform it are also known. Therefore, a type representing the robot's action given the scenario previously described is at least as follows:

$$Action = \begin{bmatrix} \text{time} & : & \mathit{Time} \\ \text{motion} & : & \begin{bmatrix} \text{distance} & : & \mathit{Real} \\ \text{heading} & : & \mathit{Real} \end{bmatrix} \\ \text{observation} & : & \mathit{RealVector} \\ \text{controls} & : & \mathit{Controls} \end{bmatrix}$$

As a result of performing an action we can read the distance in metres and the heading in degrees that the robot has travelled from the odometry component, and distances to detected points in metres from the laser scanner. A laser scanner emits a set of beams and resolves distance to points existing in space based on the known speed of light by measuring the time of flight between the camera and each point. The scanner returns a vector of distances to tracked point objects, that is, real numbers. Records corresponding to this type are not the actual objects in the world but perceptual samples of them made with some error. The type of sensors restricts what representations can be sampled. The length of the *RealVector* is dependent on the kind and the granularity of the sensor used. If a laser scanner is replaced with a video camera, for example, the observations that the robot makes will be entirely different (they will be of different type) even if the robot is observing the same world. Secondly, the granularity of the sensor determines the distinctions than can be represented with types and that can subsequently be included in other more complex record types, including those representing linguistic concepts. One can thus say that the type of the perceptual apparatus constrains the possible type representations bottom up.

After performing an action the robot can make two predictions about the type of the current state that it is in. One is based on the effects of the action that it has carried out and the other is based on what it observes around it after having carried out the action. The first of these we will call *ActionPrediction*. This is characterised by the following function:

$$\lambda \mathrm{r}_1\text{: } \mathit{RobotState}\ \lambda \mathrm{r}_2\text{: } \mathit{Action}\ \left(\begin{bmatrix} \text{pm} = f_{proc_p}(\mathrm{r}_1.\text{pm},\ \mathrm{r}_2.\text{motion}) & : & \mathit{PointMap} \\ \text{heading} = f_{proc_h}(\mathrm{r}_1.\text{heading},\ \mathrm{r}_2.\text{motion}) & : & \mathit{Real} \\ \text{time} = \mathrm{r}_1.\text{time} + 1 & : & \mathit{Time} \end{bmatrix}\right)$$

Here f_{proc_p} is a function that will use the process model of the robot to predict a new point map on the basis of the motion (adjusting only the location of the robot as everything else is assumed to be static). Similarly, f_{proc_h} is a function

that uses the robot's process model to predict a new heading on the basis of the motion.

The second prediction we will call *ObservationPrediction*. This is characterised by the following function:

$$\lambda r_1\text{: } \textit{RobotState}\ \lambda r_2\text{: } \textit{Action}\ \left(\begin{bmatrix} \text{pm} = f_{obs_p}(r_1.\text{pm},\ r_2.\text{observation}) & : & \textit{PointMap} \\ \text{heading} = f_{obs_h}(r_1.\text{heading},\ r_2.\text{observation}) & : & \textit{Real} \\ \text{time} = r_1.\text{time} + 1 & : & \textit{Time} \end{bmatrix}\right)$$

Here f_{obs_p} is a function that will predict a new point map on the basis of the observation carried out in the action (adjusting not only the location of the robot but also anything else which appears to be in a different position). Similarly, f_{obs_h} is a function that uses the robot's process model to predict a new heading on the basis of the observation.

The robot now has two potentially competing types for its new state. If the prediction about the point map and the heading are non-identical a further pair of functions must be applied to come to a final prediction, perhaps averaging the predicted point maps and headings in some way. The precise details of the nature of these functions will not concern us here. The TTR model of the update will be the same whatever the exact nature of the functions. What is important for us is that the result of the SLAM update will be a new type for the robot state. Let us refer to this state for a particular time $t+1$ as $\textit{SLAMUpdate}_{t+1}$. The complete type for the robot state at $t+1$ will be computed using the operation *asymmetric merge* ($\boxed{\wedge\!\!\!\!\cdot}$)[3]:

$$\textit{RobotState}_{t+1} = \textit{RobotState}_t \boxed{\wedge\!\!\!\!\cdot}\ \textit{SLAMUpdate}_{t+1}$$

TTR provides an account of events [13] which can be extended to the domain of robotic control. The *Action* type introduced earlier represents a type of basic event. [31] and the subsequent papers propose a theory of events in natural language semantics where events are characterised as a temporal sequence of observations of sub-events. In Fernando's theory the observations are related within a finite state automaton and hence they are treated as strings produced by a regular language. [12] introduces this view to the analysis of events in TTR by proposing a concatenation operation on types written as $T_1^\frown T_2$. This view is compatible with the description of the autonomous agent above. Let us

[3] The asymmetric merge of two types T_1 and T_2 is represented by $T_1 \boxed{\wedge\!\!\!\!\cdot}\, T_2$. If one or both of T_1 and T_2 are non-record types then $T_1 \boxed{\wedge\!\!\!\!\cdot}\, T_2$ will be T_2. If they are both record types, then for any label ℓ which occurs in both T_1 and T_2, $T_1 \boxed{\wedge\!\!\!\!\cdot}\, T_2$ will contain a field labelled ℓ with the type resulting from the asymmetric merge of the corresponding types in the ℓ-fields of the two types (in order). For labels which do not occur in both types, $T_1 \boxed{\wedge\!\!\!\!\cdot}\, T_2$ will contain the fields from T_1 and T_2 unchanged. In this informal statement we have ignored complications that arise concerning dependent types in record types. Our notion of asymmetric merge is related to the notion of priority unification [30].

represent a map building and localisation event as being of type *Exploration*. It is not known a priori how many events of the type *Action* will occur during the robot's exploration and yet such an event will still be judged as an *Exploration*. Similarly, the type *RobotState* representing the agent of the exploration event will also depend on the actions taken. An event of type *Exploration* is a string of type $RobotState^\frown(Action^\frown RobotState)^*$. However, this does not express the fact that each non-initial robot state is the result of updating the previous robot state with an action as discussed earlier. This is done by a predicate update_string with arity $[RobotState^\frown(Action^\frown RobotState)^*, RobotState{\rightarrow}(Action{\rightarrow}RecType)]$ such that update_string(s,f) is non-empty iff either $s : RobotState$ or $s = s_1^\frown a^\frown s_2^\frown s_3$, s_1:*RobotState*, a:*Action*, $s_2 : f(s_1)(a)$ and update_string($s_2^\frown s_3$,f) is non-empty. We can define a type *Exploration*:

$$\begin{bmatrix} \text{e} & : & RobotState^\frown(Action^\frown RobotState)^* \\ \text{f} & : & RobotState{\rightarrow}(Action{\rightarrow}RobotState) \\ \text{c}_{\text{update}} & : & \text{update\_string(e,f)} \end{bmatrix}$$

Preserving a direct connection between the semantics of the robotic control and natural language allows us to facilitate the mediation of information between the two, which practically proves to be a challenging task, and gives robotic control a cognitive grounding. A system using such representations could better exploit interaction in natural language with humans to learn new more complex actions from the primitive ones (explanation based learning [32]) or use knowledge expressed in types to reason about the outcomes of actions before committing to a particular one (mental simulations used for planning).

Up to now we have shown type representations of perceptual, geometric knowledge involved in spatial descriptions but not conceptual and linguistic knowledge. Geometric representations of objects introduced earlier must be given names such as 'chair' and 'table' and we also need to know their orientation determined by the front and back as this is important for projective spatial descriptions (Section 3.4). The orientation of the object can expressed as a vector by finding the central axis of an object and taking the coordinates or locations of the back-most (b) and the front-most point object (f) on this axis as its origin and endpoint respectively ($\boldsymbol{bf}$). Note that the vector is expressed as a relative displacement between two points, for example (2,3). Knowing where is the back and front is of an object is part of human knowledge about objects and their interaction, and thus part of our conceptual knowledge, just like knowing their name.

The mapping between the geometric domain and the conceptual domain is performed by classification (Section 2) which has been learned by observing a number of examples of such mapping. If o:*RegionObject* and the classifier f_{chair} applied to o.reg returns *true* then this constitutes a proof (or evidence) that o is a chair:

e:chair(o) iff $e : f_{\text{chair}}(o.\text{reg}) = 1$

We can think of the type $f_{\text{chair}}(o.\text{reg}) = 1$ as the type of event where the classifier returns 1 for o.reg. The above formulation expresses that an event of this type will constitute a proof that o is a chair. Similarly a classifier for chair orientation, $f_{\text{chair\_orient}}$, will support that the type orientation(o, R) (R:*Real*) is non-empty just in case $f_{\text{chair\_orient}}(o.\text{reg}) = R$.

With these classifications the robot created two believes about its world which must be merged to the 'beliefs' record type in the robot's state by applying the merge operation as follows: [beliefs=$T \wedge_{\cdot} [\text{c}_{\text{chair}} : \text{chair}(o)]$]. An example of a possible type for a robot which has hypothesised an object and the orientation of that object would (using p and o to represent appropriate point and object maps) be:

$$\begin{bmatrix}
\text{a} & : & \textit{Ind} \\
\text{c}_{\text{robot}} & : & \text{robot(a)} \\
\text{pm}=p & : & \textit{PointMap} \\
\text{c}_{\text{loc}} & : & \text{located(a,pm[0])} \\
\text{objects}=o & : & [\textit{Object}(\text{pm})] \\
\text{c}_{\text{object_map}} & : & \text{object_map(objects, pm)} \\
\text{beliefs}=\begin{bmatrix} \text{c}_{\text{chair}_1} & : & \text{chair}(\Uparrow\text{objects[1]}) \\ \text{c}_{\text{orient}_1} & : & \text{orientation(objects[1],}R\text{)} \end{bmatrix} & : & \textit{RecType} \\
\text{heading}=36 & : & \textit{Real} \\
\text{time}=15 & : & \textit{Time}
\end{bmatrix}$$

The '$\Uparrow$' in the constraints on the belief type indicate that the path for the arguments to the predicates are to be found in the superordinate record type.[4]

4.2 Geometric Spatial Relations

Spatial relations such as 'to the left of' hold between three objects of *RegionObject*: the located object, the reference object and the viewpoint which determines the orientation of the reference frame. The representations for intrinsic and relative perspective can be kept the same if we allow the reference object and the viewpoint be identical. On the other hand the located object and the reference object must not be identical. The mapping between objects of *RegionObject* type and a relation is done by a trained classifier. If o_1, o_2, o_3:*RegionObject*, and orientation(o_3,r:*Orientation*) and f_{left} is a left-classifier of type *Region*→*Region*→*Orientation*→*Bool*, then

e:left(o_1,o_2,o_3) iff $e : f_{\text{left}}(o_1.\text{reg}, o_2.\text{reg}, r) = 1$

[4] This is part of the abbreviatory readable notation for dependent fields in record types. Unpacking both this and the notation for manifest fields gives us the official notation:

$$\begin{bmatrix} \text{beliefs} & : & \langle \lambda v_1{:}\textit{Ind} \lambda v_2{:}\ \textit{Ind}(\textit{RecType}\begin{bmatrix} \text{c}_{\text{chair0}} & : & \text{chair(v1)} \\ \text{c}_{\text{chair1}} & : & \text{chair(v2)} \end{bmatrix}), \langle\ \text{objects[0].a, objects[1].a}\rangle\rangle \end{bmatrix}$$

That is, an event of o_1 being to the left of o_2 from the perspective of o_3 is an event of o_1's region of location being classified as to the left of o_2's region of location relative to the orientation of o_3. Two relativisations or transformations of region locations must be performed before the classification can take place: (i) the (global) coordinate frame must be rotated to correspond to the orientation of o_3; and (ii) the origin of the global coordinate frame must be transposed so that it is identical to the centre point of the region of location of o_2 [26]. Since o_1's region of location has been relativised we only need to learn one classifier function for 'left' regardless of the viewpoint.

The new belief [e:left(o_1,o_2,o_3)] is merged with the robot's beliefs. Note that we also added a belief about the robot's own identity and orientation and thus the current beliefs represent a type of situation where the chair is to the left of the table relative to the viewpoint of the robot.

$$\begin{bmatrix}
\text{a} & : & \mathit{Ind} \\
\text{c}_{\text{robot}} & : & \text{robot(a)} \\
\text{objects}=o & : & [\mathit{Object}(\text{pm})] \\
\vdots & & \\
\text{beliefs}=\begin{bmatrix}
\text{c}_{\text{me}} & : & \text{me}(\Uparrow\text{a}) \\
\text{c}_{\text{orient}_{\text{me}}} & : & \Uparrow\text{heading} \\
\text{c}_{\text{chair}_1} & : & \text{chair}(\Uparrow\text{objects[1]}) \\
\text{c}_{\text{orient}_1} & : & \text{orientation(objects[1],}R\text{)} \\
\text{c}_{\text{table}_2} & : & \text{table}(\Uparrow\text{objects[2]}) \\
\text{c}_{\text{left}_{1,2,0}} & : & \text{left}(\Uparrow\text{objects[1].a,}\Uparrow\text{objects[2].a,}\Uparrow\text{.a})
\end{bmatrix} & : & \mathit{RecType} \\
\text{heading}=36 & : & \mathit{Real} \\
\vdots &
\end{bmatrix}$$

The classifier function f_{left} represents a classification model that has been learned through observations of situations involving objects in particular geometric arrangements and human descriptions of such situations. In practical robotic applications any popular machine learning classifier could be used [33]. In [10] we used a decision tree learning method based on the ID3 algorithm and the Naïve Bayes method. The probabilistic Bayesian methods have become very popular in cognitive science as a model of human learning and cognitive development [34,35]. The benefit of the Naïve Bayes classifier is that it is simple and the reasoning with Bayesian conditional probabilities is well understood.

In Section 2 we argued that spatial descriptions are vague: they apply over a spatial continuum but to a different degree and as a result for a given configuration of a scene more than one description can hold, for example something can be 'left' and 'behind' at the same time. If there are competing descriptions possible, then one may win over another. Both facts are captured well by Bayesian classification. The probability that a situation is judged to be of type left(o_1,o_2,o_3) is estimated by observing a number of judgements events

of situations that correspond to 'left'. The judgement events may also involve other descriptions, for example 'behind', 'front', 'right', etc. If the predicate type left(o_1,o_2,o_3), behind(o_1,o_2,o_3),... is to be predicted in classification (as the target class c) from the observed geometric properties $(a_1, a_2 \dots a_n)$, according to the Bayesian rule, the relation between them is expressed as follows:

$$p(c|a_1 \dots a_i) = \frac{p(a_1, a_2 \dots a_i|v)p(c)}{p(a_1, a_2 \dots a_i)} \approx p(c) \prod_i^n \frac{p(v|a_i)}{p(a_i)}$$

In practice it is frequently naïvely assumed that the attributes are independent and hence the equation may be rewritten as on the right. In standard Naïve Bayes classification we accept the maximum a posteriori hypothesis c_{map}: the target class c with the highest probability $p(c|a_1 \dots a_i)$ and ignore other target classes. However, this does not need to be the case. One may rank the hypotheses by their probability score and then calculate a difference between the first and each subsequent candidate. If the difference between $p(c_1|a_1 \dots a_i)$ and $p(c_n|a_1 \dots a_i)$ is small and falls within some contextually determined interval $(0, 0.05)$ specified in the type representation of the discourse, both predicted ptypes are contextually applicable and hence the scene may be described as 'the chair is to the left of the table but it's also behind it'. Hypotheses with probability score deviating to a greater degree from the best candidate hypothesis are rejected as before. The probabilistic model thus accounts well how the vagueness of descriptions is resolved: through the competition of their applicability with other candidate descriptions.

4.3 Spatial Relations and World Knowledge

The world knowledge about objects, their function and typical use (functional knowledge), is expressed in types as propositional content. A situation denoted by 'the umbrella is over the man' (Section 3.3) is represented by a ptype

$$e{:}\mathrm{over}(\mathrm{o}_{\mathrm{umbrella}},\mathrm{o}_{\mathrm{man}}) \text{ iff } \begin{bmatrix} \mathrm{n} & : & \mathit{Nat} \\ \mathrm{c}_{\mathrm{rain}} & : & \Uparrow \mathrm{c}_{\mathrm{rain_n}} \\ \mathrm{o}_{\mathrm{rain}} & : & \Uparrow \mathrm{c}_{\mathrm{rain_n}}.\mathrm{rain}(\mathrm{objects}[\mathrm{n}]) \\ \mathrm{c}_{\mathrm{g\_over}} & : & \mathrm{over}_g(\mathrm{o}_{\mathrm{umbrella}},\mathrm{o}_{\mathrm{man}}) \\ \mathrm{c}_{\mathrm{protects}} & : & \mathrm{protects}(\mathrm{o}_{\mathrm{rain}},\mathrm{o}_{\mathrm{man}},\mathrm{o}_{\mathrm{umbrella}}) \end{bmatrix}$$

It involves three individuals geometrically conceptualised as volumes: the umbrella (o_1 : *VolumeObject*), the man (o_2 : *VolumeObject*) and the rain (o_3 : *VolumeObject*). Note that the third individual, the rain (o_3), is not mentioned in the description but is implied. Thus, the presence of rain is defined as an explicit constraint in the type over($\mathrm{o}_{\mathrm{umbrella}}$,$\mathrm{o}_{\mathrm{man}}$) and there is a further constraint $\mathrm{c}_{\mathrm{rain}}$ which ensures that the robot must believe that there is rain since rain must be part of its beliefs, the superordinate type of over($\mathrm{o}_{\mathrm{umbrella}}$,$\mathrm{o}_{\mathrm{man}}$) once over($\mathrm{o}_{\mathrm{umbrella}}$,$\mathrm{o}_{\mathrm{man}}$) is merged with it.

Geometrically (c_{g_over}), the umbrella must be in a particular spatial configuration with the man which is trained as a classifier. 'g_over' is typically not susceptible to perspective shifts as the viewpoint is fixed by the gravity and hence a fourth object that would determine the viewpoint is not necessary. Hence, if o_1, o_2:*VolumeObject*, and f_{over} is a over-classifier of type *Volume*→*Volume*→*Bool*, then[5]

e:g_over(o_1,o_2) iff $e : f_{g\_over}(o_1.\text{vol}, o_2.\text{vol}) = 1$

An event of o_1 being over o_2 is an event of o_1's volume being classified as 'over' o_2's volume. Before the classification takes place the origin of the global coordinate frame must be transposed to the centre point of the volume of location of o_2.

The constraint $c_{protects}$ represents conceptual knowledge in relation to ptype over($o_{umbrella}$,o_{man}) but the ptype protects(o_{rain},o_{man},$o_{umbrella}$) may be further constrained by a trained classifier. This example demonstrates how conceptual types are organised at different levels and they may be eventually dependent on perceptual classification. Language is deeply embedded in perception. An important question to address is how world knowledge constraints are acquired. With humans they may be learned by corrective feedback and explicit definition through dialogue interaction (see [17] for learning ontological conceptual knowledge this way). In a computational framework one might explore domain theory learning described in [36]. For the remaining discussion we simply assume that we have a database of predicate assertions of such knowledge.

Since the geometric meaning constraint c_{g_over} is determined by a probabilistic classifier, the acceptable deviations of the umbrella from the prototypical vertical upright position and their gradience are accounted for. Equally, the model predicts that a situation where a man holds an umbrella in the upright position and therefore the c_{g_over} constraint is defined with high probability but the umbrella does not provide protection from the rain cannot be of the ptype over($o_{umbrella}$,o_{man}) since the constraint $c_{protects}$ is not satisfied.

In Section 3.3 we discuss how world knowledge plays a greater role in the meaning representation of 'over' than 'above' which can be accounted for in a probabilistic model as follows. In Sections 4.1 and 4.2 we show that conceptual record types such as left(o_1,o_2,o_3) and chair(o) are characterised by probabilistic knowledge. This is expressing a degree of belief that particular records of perceptual types are associated with a particular record of a conceptual type: in other words that particular sensory observations map to a particular individual o of the type chair(o). The probabilistic knowledge associated with types is acquired by counting the observations events of records of appropriate type. The constraints that are a part of predicate types such as $c_{protects}$ in over($o_{umbrella}$,o_{man}) provide support for that type in the same way as perceptual observations do. They are all linked with the labels involved in the record type that contains them. The application of constraints c_1, c_2,... c_n inside a record type T may be thought of

[5] For the sake of brevity we assume that the type *Volume* type is defined similarly as the type *Region* on p.77.

as a classification function that classifies for that type T, thus $f(\pi,c_1,c_2,\ldots c_n) = \{T_1 \vee T_2, \vee T_3,\ldots, T_n\}$. The domain of the function are records of the appropriate type and the target is a type. If we induce from the records that something is of some type, then we can conclude that we have a record of this type. As a classifier the Bayes' rule may be applied.

$$p(t_{map}) \equiv \arg\max_{t \in T} p(t) \prod_{i}^{n} \frac{p(c_i|t)}{p(c_i)}$$

As before, it may happen that a particular situation may be judged nearly equally probable to belong to two more more types. The vagueness may be because the agent's knowledge is insufficient: the constraints in its types are not discriminating enough to differentiate two situations with a sufficient degree of certainty. Another reason for vagueness is that the perceptual input of the agent is insufficient to make judgements with a good degree of certainty. Introducing a contextually present interval of allowed deviation from the highest ranking probability judgement allows us to express this uncertainty linguistically, for example 'it's a chair but it could also be a sofa'.

Conceptual knowledge is the knowledge expressed in language. For purely conceptual p-types $p(t)$ and $p(c)$ can be learned by counting the number of times a record of that type is encountered in our knowledge database over the total number of predicate assertions there. ($\frac{|s_t:T_t|}{|s|}$ where s_t corresponds to a string declaration of a ptype t). In child language acquisition the knowledge database is a collection of assertions that a child hears over a period of time while interaction with a parent (the mittens-gloves example in [17]). In a computational framework this is a collection of texts in which semantic relations expressed by ptypes can be identified and extracted from (for an automatic identification of concepts see for example [37]). The probability $p(c|t)$ is equal to the number of times a particular p-type c is encountered as a defining constraint for its super p-type t in our knowledge base over the number of times the p-type c is used ($\frac{|s_c:T_c \in s_t:T_t|}{|s_c:T_c|}$). The $\frac{p(c_i|t)}{p(c_i)}$ from the body of the Bayes' rule above defines the support of c for t. Its value will be maximum (1) if c exclusively constrains t and none of the other types.

For perceptual p-types c_k, which represent both continuous numeric knowledge and conceptual knowledge, the probability of a record belonging to the type $p(c_k)$ is determined by the application of a probabilistic classifier which links the two domains (Section 4.2, p.84). The subscript k indicates a specific ptype returned by the perceptual classifier. The probability $p(c_k|t)$ is determined by observing the number of times the predicted ptype c_k is used as a constraint for its super p-type t over the number of times c_k is used in general ($\frac{|s_{c_k}:T_{c_k} \in s_t:T_t|}{|s_{c_k}:T_{c_k}|}$). Consider the case of spatial descriptions 'over' and 'above'. In the learning stage of the type *Over* one may observe assignments to this type where the geometric constraint is ignored in favour of the conceptual constraint. Consequently, the support of the geometric constraint c_{k_1} for t is lower and that of the conceptual constraint c_2 is higher. The reverse is the case for the type *Above*. Therefore, the

probabilistic model correctly predicts the facts observed experimentally. Overall, the preceding discussion shows how a probabilistic model of TTR provides the required information fusion between the continuous numeric and conceptual domains.

5 Conclusion and Future Work

In this paper we present a formal semantic model for natural language spatial descriptions in Type Theory with Records which can incorporate perceptual, conceptual and discourse constraints. In particular we discuss perceptual and conceptual constraints. The TTR framework combines many familiar notions such as semantic compositionality, logical reasoning, feature structures, unification, (probabilistic) classification, dynamic models that are present in other frameworks but without the synergies allowed for by this framework. Independently of our treatment of the semantics of spatial descriptions the notion of judgement (which practically corresponds to perceptual classification) is at the very heart of the theory and it is in this respect the theory is most distinct from standard model-theoretic theories of natural language semantics in the Montagovian tradition. These assume that meaning is determined through a denotation function which checks the assignment of expressions to a model of the world. Practically, this poses at least two problems. The exact mechanics of this functions are not accounted for as they are not considered a part of the theory. However, such a theory can not model language that is grounded in perception and appropriate extensions are required. Secondly, it is not clear how an agent can have access to a complete representation of the world in order to be able to verify the meanings of expressions against it. Even if a representation of such a world is pre-given to the agent, the agent would not be able to track the changes in the world that happen independently of it. The TTR does not presuppose a complete model of the world which is directly accessible to the agent, but instead it assumes that an agent is situated in a local sub-part of such a world with which it can interact and from which it can take samples of observations. The interaction can be linguistic (conversations with other agents) or perceptual (sensory observations). The meaning is resolved through a judgement that a particular situation refers to a particular type. A judgement only holds for a particular agent and its particular interaction history. That different agents nonetheless build very similar though not quite identical representations and are able to communicate with each other and their environment is due to their capacity to make refinements of the model from the feedback they get from the environment continuously. The notion of meaning in the TTR framework is therefore dynamic. Overall, this move from fixed meaning representations to dynamic meaning representations has also been witnessed in the transition from early robotics to modern robotics employing machine learning. A reliable agent is the one that continuously adapts to the changes in its environment.

The preceding discussion demonstrates how one can gradually abstract from perception to concept formation to language in a single framework of types where

there is no sharp boundaries between the domains, just increased complexity of types. In this respect TTR does not distinguish between symbolic and sub-symbolic domains that are common in the traditional AI approaches. Instead, we refer to low-level and high level types which contain conceptual or distributional information or both. The sensory apparatus of an agent constraints the type representations that it can form bottom up since the simple types are those that correspond to the quantities that can be sampled from the real world. An agent can then use the compositional properties of the type system and probabilistic classifiers to build increasingly more complex types, points from sets of sensory observations, regions from sets of points, etc. that are common in human cognition. Since objects can be represented as different geometric types, spatial relations these these types as their arguments will also have multiple type representations. This is confirmed by our intuition since 'left' in 2-dimensional space is a distinct relation (or type) from 'left' in 3-dimensional space.

Since not all agents have the same kinds of sensors, for example humans and robots, it follows that the type representations that they build may be also considerably different. For example, if a 3-dimensional laser scanner were employed in localisation and map building, the resulting types representing the same objects would contain different labels and different constraints would be formed over them. Strictly speaking, even if two agents have the same sensory systems and since probabilistic learning is involved, they would not learn exactly the same representations since they would not have observed the same situations both in type and number. An interesting observation is that in spite of such differences in their conceptual systems, the agents are still able to communicate with each other and relate to the same physical world. As we have seen with the localisation procedure continuous updating and improving the representations through interaction is crucial for making the model more and more accurately represent the world. The same also holds for negotiating concepts through dialogue interaction with humans [19].

In the future we are intending to implement the current proposal computationally in a robotic agent. Here we foresee the following challenges: (i) how can we ensure the same kind of compositionality of perceptual meaning that we get with conceptual meaning; (ii) how do we learn the right kind of world knowledge relevant for a particular spatial description; and (iii) how does such world knowledge link to perception given that it has been learned from assertions and has never been experienced through sensory observations. To address the second issue we are exploring two strategies: (a) mining the knowledge about objects that occur with a particular spatial description from a large text corpora, and (b) using a variant of explanation based learning in an inquisitive dialogue agent that learns the conceptual constraints from a conversation with a human dialogue partner.

Acknowledgements. This research was supported in part by VR project 2009-1569, Semantic analysis of interaction and coordination in dialogue (SAICD) and Centre for Language Technology at University of Gothenburg, Sweden.

References

1. Blackburn, P., Bos, J.: Representation and inference for natural language. A first course in computational semantics. CSLI (2005)
2. Winograd, T.: Understanding Natural Language. Edinburgh University Press (1976)
3. Miller, G.A., Johnson-Laird, P.N.: Language and perception. Cambridge University Press, Cambridge (1976)
4. Gapp, K.P.: Basic meanings of spatial relations: Computation and evaluation in 3d space. In: Hayes-Roth, B., Korf, R.E. (eds.) AAAI, pp. 1393–1398. AAAI Press/The MIT Press (1994)
5. Logan, G.D., Sadler, D.D.: A computational analysis of the apprehension of spatial relations. In: Bloom, P., Peterson, M.A., Nadel, L., Garrett, M.F. (eds.) Language and Space, pp. 493–530. MIT Press, Cambridge (1996)
6. Siskind, J.M.: Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. Journal of Artificial Intelligence Research 15, 31–90 (2001)
7. Roy, D.: Semiotic schemas: a framework for grounding language in action and perception. Artificial Intelligence 167, 170–205 (2005)
8. Guerra-Filho, G., Aloimonos, Y.: A language for human action. Computer 40, 42–51 (2007)
9. Matuszek, C., Herbst, E., Zettlemoyer, L., Fox, D.: Learning to parse natural language commands to a robot control system. In: Proc. of the 13th International Symposium on Experimental Robotics, ISER (2012)
10. Dobnik, S.: Teaching mobile robots to use spatial words. PhD thesis, University of Oxford: Faculty of Linguistics, Philology and Phonetics and The Queen's College, Oxford, United Kingdom (2009)
11. Cooper, R.: Austinian truth, attitudes and type theory. Research on Language and Computation 3, 333–362 (2005)
12. Cooper, R.: Type theory and semantics in flux. In: Kempson, R., Asher, N., Fernando, T. (eds.) Handbook of the Philosophy of Science. General editors: Gabbay, D.M., Thagard, P., Woods, J., vol. 14. Elsevier BV (2012)
13. Davidson, D.: The logical form of action sentences. In: Rescher, N., Anderson, A.R. (eds.) The Logic of Decision and Action. University of Pittsburgh Press, Pittsburgh (1967)
14. Barwise, J.: The situation in logic. CSLI Publications, Stanford (1989)
15. Cooper, R.: Records and record types in semantic theory. Journal of Logic and Computation 15, 99–112 (2005)
16. Larsson, S., Cooper, R.: Towards a formal view of corrective feedback. In: Alishahi, A., Poibeau, T., Villavicencio, A. (eds.) Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition, EACL, pp. 1–9 (2009)
17. Cooper, R., Larsson, S.: Compositional and ontological semantics in learning from corrective feedback and explicit definition. In: Edlund, J., Gustafson, J., Hjalmarsson, A., Skantze, G. (eds.) Proceedings of SemDial 2009 (DiaHolmia): Workshop on the Semantics and Pragmatics of Dialogue, Stockholm, Sweden, Royal Institute of Technology (KTH), pp. 59–66 (2009)
18. Ginzburg, J.: The interactive stance: meaning for conversation. Oxford University Press, Oxford (2012)

19. Larsson, S.: The TTR perceptron: Dynamic perceptual meanings and semantic coordination. In: Artstein, R., Core, M., DeVault, D., Georgila, K., Kaiser, E., Stent, A. (eds.) SemDial 2011 (Los Angelogue): Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue, Los Angeles, California, pp. 140–148 (2011)
20. Farhadi, A., Endres, I., Hoiem, D., Forsyth, D.: Describing objects by their attributes. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1778–1785 (2009)
21. Herskovits, A.: Language and spatial cognition: an interdisciplinary study of the prepositions in English. Cambridge University Press, Cambridge (1986)
22. Talmy, L.: Toward a cognitive semantics: concept structuring systems, vol. 1 and 2. MIT Press, Cambridge (2000)
23. Regier, T., Carlson, L.A.: Grounding spatial language in perception: an empirical and computational investigation. Journal of Experimental Psychology: General 130, 273–298 (2001)
24. Coventry, K.R., Prat-Sala, M., Richards, L.: The interplay between geometry and function in the apprehension of Over, Under, Above and Below. Journal of Memory and Language 44, 376–398 (2001)
25. Levinson, S.C.: Space in language and cognition: explorations in cognitive diversity. Cambridge University Press, Cambridge (2003)
26. Maillat, D.: The semantics and pragmatics of directionals: a case study in English and French. PhD thesis, Committee for Comparative Philology and General Linguistics, University of Oxford, Oxford, UK (2003)
27. Steels, L., Loetzsch, M.: Perspective alignment in spatial language. In: Coventry, K.R., Tenbrink, T., Bateman, J.A. (eds.) Spatial Language and Dialogue. Oxford University Press (2009)
28. Dobnik, S.: Spatial descriptions in discourse: choosing a perspective. In: Brown-Schmidt, S., Jonathan Ginzburg, S.L. (eds.) Proceedings of SemDial 2012 (SeineDial): The 16th Workshop on the Semantics and Pragamtics of Dialogue, Université Paris-Diderot (Paris 7), Paris Sorbonne-Cité, pp. 151–152 (2012)
29. Dissanayake, M.W.M.G., Newman, P.M., Durrant-Whyte, H.F., Clark, S., Csorba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotic and Automation 17, 229–241 (2001)
30. Shieber, S.: An Introduction to Unification-Based Approaches to Grammar. CSLI Publications, Stanford (1986)
31. Fernando, T.: A finite-state approach to events in natural language semantics. Journal of Logic and Computation 14, 79–92 (2004)
32. Lauria, S., Bugmann, G., Kyriacou, T., Bos, J., Klein, E.: Training personal robots using natural language instruction. IEEE Intelligent Systems 16, 38–45 (2001)
33. Witten, I.H., Frank, E., Hall, M.A.: Data mining: practical machine learning tools and techniques, 3rd edn. Morgan Kaufmann, Burlington (2011)
34. Clark, A., Lappin, S.: Linguistic nativism and the poverty of the stimulus. Wiley-Blackwell, Chichester (2011)
35. Tenenbaum, J.B., Kemp, C., Griffiths, T.L., Goodman, N.D.: How to grow a mind: Statistics, structure, and abstraction. Science 331, 1279–1285 (2011)
36. Pulman, S.G., Liakata, M.: Learning domain theories. In: Nicolov, N., Bontcheva, K., Angelova, G., Mitkov, R. (eds.) RANLP. Current Issues in Linguistic Theory (CILT), vol. 260, pp. 29–44. John Benjamins, Amsterdam/Philadelphia (2003)
37. Baroni, M., Murphy, B., Barbu, E., Poesio, M.: Strudel: A corpus-based semantic model based on properties and types. Cognitive Science 34, 222–254 (2010)