A Types-As-Classifiers Approach to Human-Robot Interaction for Continuous Structured State Classification

Julian Hough
Cognitive Science Group
School of Electronic Engineering and Computer Science
Queen Mary University of London
j.hough@qmul.ac.uk

Lorenzo Jamone
Centre for Advanced Robotics
School of Electronic Engineering and Computer Science
Queen Mary University of London
l.jamone@qmul.ac.uk

David Schlangen
Foundations of Computational Linguistics Lab
Linguistics Department
University of Potsdam
david.schlangen@uni-potsdam.de

Guillaume Walck
Neuroinformatics Group
Faculty of Technology
Bielefeld University
gwalck@techfak.uni-bielefeld.de

Robert Haschke
Neuroinformatics Group
Faculty of Technology
Bielefeld University
rhaschke@techfak.uni-bielefeld.de

Abstract

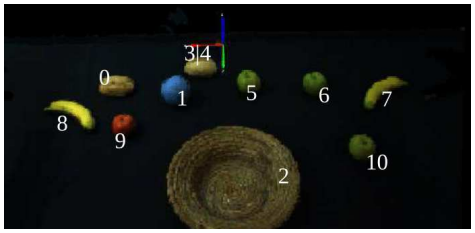
While flat representations of dialogue states can be useful for machine learning approaches to human-robot interaction, there is still a role for structured dialogue states classification, particularly for domains with little data. To address this, we propose a novel types-as-classifiers approach to dialogue processing for robots using probabilistic type judgements. In our proposal, incoming sensory data is converted to a world belief Type Theory with Records (TTR) record type in real time, and then derived beliefs such as intention attribution to a user, or the prediction of the affordances of visible objects, are made as record type judgements of that record type. The world belief record type can be updated dynamically like a dialogue state, allowing information of different perceptual sources to be easily combined using simple composition mechanisms using standard probability theoretic axioms.

1 Introduction

The combination of computer vision and natural language processing is now popular. Thanks to increased computing power and the development of new deep learning techniques, huge strides forward have been made in several tasks, including: automatic image retrieval from key words, reference resolution of objects in photographs from textual referring expressions (Kennington and Schlangen, 2015), generating referring expressions to objects given probabilistic estimation of object properties (Mast et al., 2016), caption generation and visual question answering (Antol et al., 2015).

A more challenging task, beyond the use of single sentences with images, is designing dialogue systems for real-world human-robot interaction (HRI) which combine probabilistic information encoding visual and physical properties of objects and information about the interaction which a dialogue system would encode in a dialogue state. This uniform approach not only requires the use of complex visual information and semantic parsing, but also needs to permit fluid interaction with a collaborative robot to help a user complete a manual task. This requires an incrementally and dynamically evolving dialogue state which encodes the robot's own action state as well as its estimation of the user's intentions in real time. While flat structures can be used to encode dialogue system states, to cover relations between objects and hierarchical robot states, particularly when only a small amount of training data is available, hierarchical structure can help as a starting point for more efficient learning and greater flexibility.

SCENE:



OBJECTS (segmentation and visual classifiers):

obj_0:
yellow = 0.9627010226249695
blue = 0.0000065658565517
..
position_x = 349.3824768066406
position_y = 230.4832458496094
position_z = 21.07515907287598

obj_1:
yellow = 0
blue = 0.9758355617523193
...
position_x = 521.5785522460938
position_y = 405.300048828125
position_z = 42.72132110595703

...

USER SPEECH (current user utterance):
'put the left green apple in the basket'

ROBOT ACTION AND TASK STATE:

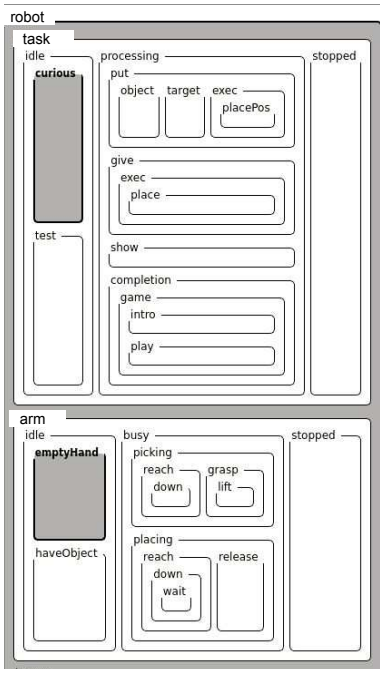


Figure 1: A typical state according to the robot. Objects are segmented and properties can be obtained for each object. The robot's internal action state is controlled by a Hierarchical State Machine (HSM)

In this paper we address this challenge by formulating a simple interaction state for a manipulator robot with natural language understanding capability using concepts from Type Theory with Records (TTR) (Cooper, 2005). We characterize the robot's world belief as a constantly updating *record type*, and use type classifiers of different kinds which operate on the state record type to make type judgements on the world belief. Once a judgement is made, this can be added to the world belief for further classification and update. Our approach allows a variety of different classification techniques to be used, but for classifier composition we use a combination of lattice theory and probabilistic TTR (Cooper et al., 2014). Inspired by the recent work using TTR for perceptual classification (Dobnik et al., 2012; Yu et al., 2016; Larsson, 2018) and Kennington and Schlangen (2015)'s simple and elegant words-as-classifiers model to reference resolution of objects in real-world scenes, here we propose a more general types-as-classifiers approach to interactive robots with natural language understanding capability.

For the remainder of the paper we give the technical backbone to the types-as-classifiers approach in Section 2 and distinguish two different types of classifier and explain them, namely *extensional* classifiers in Section 3 and *intensional* classifiers in Section 4. In Section 5 we show a detailed example application in a real-world scenario and discuss how our system can deal with ambiguity and conclude in Section 6.

2 Types-As-Classifiers for human-robot interaction

For the kinds of robot we are concerned with, namely collaborative pick-and-place robots, an example snapshot of the robot's internal state in terms of its incoming raw perceptual input is as in Fig. 1. The left side shows a camera feed, and computer vision based segmentation and tracking of objects as described by Ückermann et al. (2014a,b). The example also displays the x,y and z coordinates for the centroid of the position of the objects, and the results of real-valued perceptual classifiers applied to each object, such as that for 'yellow', classifying the degree to which an object has a particular perceptual property in the range $[0, 1]$– while these can be taken as raw input to our system, a types-as-classifiers foundation for these will be explained below in Section 3.1. The current words recognized by the robot's automatic speech recognizer (ASR) are also added to the state as they arrive. On the right side, the diagram shows how the robot tracks its own current task state and action state of its arm through a Hi-

erarchical State Machine (HSM), where the dark areas are currently active states.

2.1 Probabilistic TTR

In this paper we use TTR *record types* as the principle mathematical object of interest. We will briefly overview TTR, though see Cooper (2005) for details. Each record type consists of a set of *fields*, where each field consists of a pair of a *label* and a *type* and is notated $l : T$ denoting the judgement that an object labelled l is of type T, where T can be either an atomic type, a predicate type with arguments of other typed objects, or an embedded record type. All types are of type $Type$ (including record types), and the whole type lattice is ordered by the subtype relation $\sqsubseteq$, has the meet relation $\wedge$ (*merge* operation, union of fields for record types) and the join relation $\vee$ (*minimal common supertype*, intersection of fields for record types) and with these two relations, they obey the laws of idempotency, commutativity, associativity, absorption, and distributivity (Hough and Purver, 2017).

For probabilistic type judgements following Cooper et al. (2014), the probability judgements of the form $p(a : T)$ are the real-valued probability that object a is of type T. For record type judgements, the standard product rule and Bayes' rule hold using the $\wedge$ operation in place of a conjunction, and the sum rule holds using disjunction of types (though the disjunction of types is not equivalent to the $\vee$ relation)– see Hough and Purver (2017) for details.

2.2 Encoding the robot's sensory state as an updating TTR record type

Key to our types-as-classifiers approach is encoding the robot's current internal state as a record type which can then undergo further type judgements. We characterize the perceived state of the robot in the interaction as a *world belief* record type wb– for an in-robot control system for our purposes it will be of the format in (1).[1]

$$wb : \left[\begin{array}{lll} objects & : & \left[\begin{array}{ll} obj\_0 & : \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \\ obj\_1 & : \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \\ \ldots & : & \ldots \\ obj\_n & : \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \end{array}\right] \\ robot & : & \left[\begin{array}{lll} arm & : & \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \\ task & : & \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \\ intention & : & \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \end{array}\right] \\ human & : & \left[\begin{array}{ll} c{-}utt & : \left[\begin{array}{ll} parse & : \ldots \\ words & : \ldots \end{array}\right] \\ status & : \ldots \\ intention & : \left[\begin{array}{lll} \ldots & : & \ldots \end{array}\right] \end{array}\right] \end{array}\right] \quad (1)$$

For HSMs as in the right-hand side of Fig. 1, we can formulate the state at a given time as a record type with a recursive structure. The record type gets constructed from the highest level downwards, whereby each parallel, concurrent state, such as the *task* and *arm* sub-states of *robot* in Fig. 1, are encoded as separate sister fields in the record type. If the current active state is an embedded substate, for example the *emptyHand* and *holdsObject* substates within the *idle* substate of the *arm* state in Fig. 1, that will be encoded in the record type structure as an embedded record type (a record type within a record type). When a given field in the state has a value which is non-decomposable or 'atomic', that will be encoded as a single value in the record type with no further sub-record type. Using this recursive formulation, the robot's current action and task state in the example snapshot, shown by the darkened areas in Fig. 1, can be formulated as in (2). This is an efficient way of encoding this part of the state, as the inactive substates as shown in the statechart need not be encoded explicitly in state updates.

$$\left[\begin{array}{ll} robot & : \left[\begin{array}{ll} task & : \left[\begin{array}{ll} idle & : curious \end{array}\right] \\ arm & : \left[\begin{array}{ll} idle & : emptyHand \end{array}\right] \end{array}\right] \end{array}\right] \quad (2)$$

The continuous, incremental interpretation process of our system is a probabilistic state update, whereby wb is updated using a conditional probability judgement at each time-step. This judgement is the likelihood that wb at time t is of record type i from within a set of possible disjunctive (mutually exclusive, or clashing) record types I, conditioned by evidence record type e from the last recorded time-step $t{-}1$. In a traditional machine learning classification set-up e can be seen

[1] This is an example record type where many of the labels and values are just represented by '...' to indicate at least one such field would be present in the full representation.

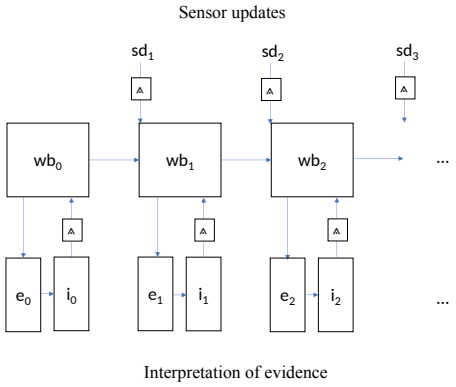


Figure 2: Illustration of the continuous world belief update process. Sensor updates sd update the previous current world belief wb_{i-1}, then evidence type e is made of wb_i which is then classified as record type i, which is then merged into override matching fields in wb_i.

as the 'instance' of data being classified, however here we assume that e constitutes part of (a *supertype* of) wb which is independent of the rest of wb such that the resulting judgement of e applies to the whole of wb– that is to say the judgement is incremental in the sense of Sundaresh and Hudak (1991) where the part is classified independently without affecting the rest of the record type.

The core process of interest is classification followed by a dynamic update to wb using the output of that classification. The new classifications are triggered by a new incoming sensor update sd_t, whose field values override the corresponding ones in wb from the previous time-step (so at this point in the update process sd_t will now be a supertype of wb_t). The classification then takes place on this updated wb where new type judgement override the old ones. Formally, the general update procedure is therefore as in the two steps in (5), where $\boxed{\mathbin{\mathbb{A}}}$ is TTR *asymmetric merge* (Dobnik et al., 2012; Hough, 2015).[2] The update dynamics to wb over time using these two recurrent steps can be seen illustratively as in Fig. 2.

$$\begin{aligned} &1.\ wb_t := \begin{cases} sd_t & \text{if } t = 0 \\ wb_{t-1} \boxed{\mathbin{\mathbb{A}}} sd_t & \text{otherwise} \end{cases} \\ &2.\ wb_t := wb_t \boxed{\mathbin{\mathbb{A}}} \arg\max_{i \in I} p(wb_t : i | wb_t : e) \end{aligned} \tag{5}$$

Note the time-step subscripts will be suppressed from here onwards, as they do not add any useful information in explaining the update process, but they are included here to make it explicit that the classification for the current state is done based on the last state that is recorded.

In Sections 3-4 we outline different perceptual classifiers which operate on different values for e (supertypes of wb relating to different parts of it) to get the conditional probability judgement that wb is of a given type. In Section 5 we will show how this can be done recursively– once a type judgement is made (by a particular type of classifier). The way in which the set I is defined for a given conditioning RT e, and the conditional probability value for each i in I is calculated depends on the classifier being used. Before we outline those specific classifiers we give the technical background to the composition of classifiers and how probabilistic functions are used.

2.3 Composing classifiers and independence assumptions

With a number of different types of classifier at our disposal as will be described below, the state is updated as they are applied to and update wb according to the update protocol in (5). Depending on the type of classifier, the probability values of their application are computed in different ways. We are first concerned with what we will call *extensional* type classifiers, those from independent classification judgements from the real world sense data

[2]The asymmetric merge operator returns the union of the fields of an ordered pair of two RTs, but where there are clashing field values for the same field label, the value from the RT on the right-hand side of the operator take precedence over the left-hand RT. Formally, this is as follows for two RTs L and R, where the $-$ sign is set difference, whereby for any two sets S and T, S–$T = \{s \mid s \in S \text{ and } s \notin T\}$:

$$L \boxed{\mathbin{\mathbb{A}}} R = (fields(L) - fields(R)) \cup fields(R)$$

$$p(r:\begin{bmatrix} x & : & e \\ A & : & A(x) \\ B & : & B(x) \end{bmatrix}) = p(r:\begin{bmatrix} x & : & e \\ A & : & A(x) \end{bmatrix} \mid r:\begin{bmatrix} x & : & e \\ B & : & B(x) \end{bmatrix}) \times p(r:\begin{bmatrix} x & : & e \\ B & : & B(x) \end{bmatrix})$$

$$= p(r:\begin{bmatrix} x & : & e \\ B & : & B(x) \end{bmatrix} \mid r:\begin{bmatrix} x & : & e \\ A & : & A(x) \end{bmatrix}) \times p(r:\begin{bmatrix} x & : & e \\ A & : & A(x) \end{bmatrix}) \quad (3)$$

$$p(r:\begin{bmatrix} x & : & e \\ A & : & A(x) \\ B & : & B(x) \end{bmatrix}) = p(r:\begin{bmatrix} x & : & e \\ A & : & A(x) \end{bmatrix}) \times p(r:\begin{bmatrix} x & : & e \\ B & : & B(x) \end{bmatrix}) \quad (4)$$

Figure 3: Product Rule for classifiers– the general rule in (3) and the rule for independent classifiers in (4).

sd. We do not deal with extensional type judgements which are dependent on one another in this paper, but, as shown in Fig. 3, consistent with standard probability theory, the general *product rule* for two classifiers A and B being applied to the same instance x within a record type is as in (3), and if A and B are independent of each other, as we assume of the extensional classifiers in this paper, the product of their probability is calculated by simple multiplication as in (4). Furthermore, if two types T_1 and T_2 are not dependent on each other, including if they are record types, then we also assume independence as in (6):

$$p(r:\begin{bmatrix} r_2 & : & T_2 \\ r_1 & : & T_1 \end{bmatrix}) = p(r:[\ r_1 : T_1\ ]) \times p(r:[\ r_2 : T_2\ ]) \quad (6)$$

2.4 Type Function classifiers

Before explaining the probabilistic type functions, the non-probabilistic type function we assume is a mapping of the judgement an object is of a given type to the judgement of it being of a (possibly different) type. For some type function $\lambda x : T_d.x : T_r$ we assume we have a set of domain types which are all a subtype of some type T_d and a range type T_r, so that for some object r, an application of $\lambda x : T_d.x : T_r$ gives (7) :

$$(\lambda x : T_d.x : T_r)(r) = \begin{cases} r : T_r & \textit{iff}\ \ r \sqsubseteq T_d \\ abort & otherwise \end{cases} \quad (7)$$

To generalize this to probabilistic type functions we enhance this with a probability function δ^{T_d}, with a similar function to a *conditional probability table* in Bayesian networks, assigning the conditional probability value to the range type T_r given the domain input T_d. These assignments are consistent with the lattice-theoretic properties of type lattices observed by Hough and Purver (2017). This gives the formulation in (8):

$$p((\lambda x : T_d.x : T_r)(r)) = \begin{cases} p(r : T_r) = \delta^{T_d}(T_r) & \textit{iff}\ \ r \sqsubseteq T_d \\ 0 & otherwise \end{cases} \quad (8)$$

For example, take object r as being judged to be a subtype of $grassWet$, and we want to get the resulting type judgement and probability of the function $\lambda x : grassWet.x : rained$ being applied to r, given the probability distribution $\delta^{grassWet}$ is as follows:

	$rained$	$\neg rained$
$grassWet$	0.7	0.3

Given r is a subtype of $grassWet$, the resulting probability judgement after application of the function would be $p(r : rained) = 0.7$.

This simple formulation is sufficient for our purposes. The domain and range types will be complex, record types (denoted short hand in the above), but all rely on the subtype checking, which, if passed, give a probability judgement of the range type.

In the following sections we will present the different type classifiers and type function classifiers we use in our system, explaining how the probabilities are computed. While we suggest a pipeline here by presentation order, we are not committed to a specific classification ordering or algorithm for inter-leaving these processes, using a simple method here, and leaving investigation into alternatives for future work.

3 Extensional classifiers

First we consider *extensional* classifiers, those that directly apply to the incoming sensory data sd, we

consider judgements on objects in the visual scene and also the action state of the robot (i.e. in our case the position of the arm).

3.1 Grounding atomic type judgements on sensory data

While as in Fig. 1 we show example inputs already at the level of basic type judgements on raw input data, we outline briefly how the lowest level classifiers can characterized in our types-as-classifiers framework. *Atomic* probability judgements from the sensory data, such as those single type judgements on single objects in the visual scene, can be either discriminative or generative classifiers which extract features from objects with a feature vectorizer function $feat$. For example, a logistic regression classifier which yields the degree between $[0, 1]$ which an object x is classified as blue by the classifier c^{LR}_{blue}, where objects in question have m features, uses the record type in (9), where β_0 to β_m are coefficients, with β_i for $i \geq 1$ corresponding to features yielded from the $feat$ function.

$$p(r : \begin{bmatrix} x & : e \\ c_{blue} & : blue(x) \end{bmatrix}) = p(r : \begin{bmatrix} x & : e \\ f & : feat(x) \\ \beta_0 & : \mathbb{R} \\ \beta_1 & : \mathbb{R} \\ \dots & : \dots \\ \beta_m & : \mathbb{R} \\ c^{LR}_{blue} & : blue(f, \beta_0 \dots \beta_m) \end{bmatrix}) = \frac{1}{1 + e^{-(r.\beta_0 + \sum_{i=1}^{m} r.\beta_i r.f_i)}} \quad (9)$$

An equivalent classification formula for a Naive Bayes classifier for blue c^{NB}_{blue} would be as follows:

$$p(r : \begin{bmatrix} x & : e \\ c_{blue} & : blue(x) \end{bmatrix}) = p(r : \begin{bmatrix} x & : e \\ f & : feat(x) \\ c^{NB}_{blue} & : blue(f) \end{bmatrix}) = p(r : B) \prod_{i=1}^{m} p(f_i \in r.f \mid r : B) \quad \text{where } B = \begin{bmatrix} x & : e \\ c_{blue} & : blue(x) \end{bmatrix} \quad (10)$$

We note we could also scale this to neural classifiers, but for now we are concerned with classifiers with more readily interpretable models which allow relatively simple modes of composition.

3.2 Grounding classifiers to sets of objects

While the classifiers just explained apply to single objects, in this paper we deal with plurals and quantification, allowing multiple objects to be referred to. When a type judgement applies to a set of objects, we assume independence and use the product of the probability of each member of the set being of a given type. Here we also introduce the notion of the type judgement being *grounded* into the set of objects, in line with the natural language grounding motivations (Roy, 2005; Larsson, 2018). To denote grounding predicate type judgements, we introduce the predicate $G(a, s)$ which means for a given type a and given set of perceived objects or events s, we are judging those objects to be of type a, i.e. grounding them. The full set classifier is as in (11):

$$p(r : \begin{bmatrix} s & : set \\ a & : Type \\ g & : G(a, s) \end{bmatrix}) = \prod_{obj \in r.s} p(obj : r.a) \quad (11)$$

An example usage of this grounding classifier for the object set $\{obj\_1, obj\_2\}$ is as follow for the joint likelihood of both objects in the set being classified as blue. Note we do not commit to the lower level classification method of the objects here, as it could be a variety of discriminative or generative classifiers as exemplified in (9) or (10):

$$p(r : \begin{bmatrix} s & : \{obj\_1, obj\_2\} \\ a & : \begin{bmatrix} x & : e \\ c_{blue} & : blue(x) \end{bmatrix} \\ g & : G(a, s) \end{bmatrix}) = \prod_{obj \in r.s} p(obj : r.a) \quad (12)$$

3.3 Complex relational extensional classifiers for relative position

Complex sensory type classifiers which take arguments such as 'x to the left of y' are also extensional, as in their input is directly from the sensory data, but they take multiple inputs. Here we simply use the concatenation of the feature vectors from the two objects involved into f, meaning the use of $left\_of$ applied to two objects x and $x1$ in the logistic regression classifier is in (13).

$$p(r : \begin{bmatrix} x & : e \\ x1 & : e \\ c_{lo} & : left\_of(x, x1) \end{bmatrix}) = p(r : \begin{bmatrix} x & : e \\ x1 & : e \\ f & : feat(x) \oplus feat(x1) \\ \beta_0 & : \mathbb{R} \\ \beta_1 & : \mathbb{R} \\ \dots & : \dots \\ \beta_m & : \mathbb{R} \\ c^{LR}_{lo} & : left\_of(f, \beta_0 \dots \beta_m) \end{bmatrix}) = \frac{1}{1 + e^{-(r.\beta_0 + \sum_{i=1}^{m} r.\beta_i r.f_i)}} \quad (13)$$

We do not commit to the $feat$ function for relative position classification only using spatial features, as we would hope the relevant features would be learned, as was shown successfully in

$$p(wb : \left[objects.obj\_1 : \left[\begin{array}{ll} x & : e \\ c\_graspable & : graspable(x) \end{array} \right] \right] \mid wb : \left[objects.obj\_1 : \left[\begin{array}{ll} pos.x & : 145 \\ pos.y & : 499 \\ pos.z & : 303 \end{array} \right] \right]) = 0.57$$

Figure 4: A conditional record type judgement involving the affordance judgement of how graspable an object is.

the equivalent words-as-classifiers models for spatial descriptions using logistic regression classifiers (Kennington and Schlangen, 2015) and also the perceptron classification approach to position classification by Larsson (2015).

3.4 Object affordance classification

Before we go on to describe intention recognition, a pre-intentional classification of the situation is the robot's perception of object properties based on incoming sensory information, which is vital for complex interaction with the human user. Particularly for manipulator robots, the perception of object *affordances* (Gibson, 2014), i.e. the possible actions associated to the objects (e.g. *graspable*), is crucial for the robot to be able to manipulate them (Jamone et al., 2016). Recently, probabilistic computational models of affordance perception have been proposed, using Bayesian Networks (Gonçalves et al., 2014) and variational auto-encoders (Dehban et al., 2016)- these can be used to obtain the probability of an object having different affordances from visual and linguistic features. In our model, affordance prediction is part of the probabilistic type judgement of wb, such that the probabilities of each object having each affordance property are part of the available type judgements.

We make no commitment to a particular model, though Gonçalves et al. (2014)'s Bayesian network approach is most easily intergrated into our model here. An example of the probabilistic judgement involved would be as in Fig. 4.

4 Intensional type classifiers

We now describe *intensional* classifiers whose probability values on application are derived from the lower-level classifiers described in the previous section. For the natural language understanding part of the system, classifiers are used according to the structures produced by the parser, which will be briefly described first, and then used to classify the user's intention by grounding type judgements into user intention record types such as i in Fig. 5. The human intentions our simple robot computes consist simply of the $action$ type, the $objects$ being manipulated, and the $goal$, which encodes the end target location of the objects, further specified by a landmark set of object lm and a relative location of the target to that landmark rel_loc.

4.1 Parsing

The record types from the $human.c{-}utt.parse$ field of the world belief record type wb are populated by the Dylan ('DYnamics of LANguage') parser (Purver et al., 2011).[3] The parser fulfills the criteria for incremental semantic construction outlined in Hough et al. (2015): it consumes words one-by-one and outputs a maximal semantic record type (RT) based on a pre-defined Dynamic Syntax-TTR (DS-TTR) grammar– see Eshghi et al. (2011) for full details. The types from the parse are entities e, predicate types t, events es and integers $\mathbb{N}$. A parse for 'put the red apple in the big basket' is in Fig. 6.

4.2 Incremental intention classification including grounded reference resolution

As DyLan's DS-TTR parser provides RTs word-by-word incrementally, the user's intention can also be estimated word-by-word as wb is updated in this fine-grained manner. Given a set of possible user intention record types I, where a typical intention may look like i in Fig. 5, and the conditioning evidence e, a record type representing a sub-part of wb as described above, we characterize a standard Maximum Likelihood multi-class probabilistic classifier to estimate the best prediction for the $human.intention$ field and its probability (or *confidence*) in its prediction $Ev(human.intention)$ by the standard $arg\ max$

[3] Available from `https://bitbucket.org/dylandialoguesystem/dsttr`

$$i = \left[\begin{array}{l} human : \left[\begin{array}{l} intention : \left[\begin{array}{ll} goal & : \left[\begin{array}{ll} lm & : \{obj\_2\} \\ rel\_loc & : INTO \end{array} \right] \\ objects & : \{obj\_9\} \\ action & : PUT \end{array} \right] \end{array} \right] \end{array} \right]$$

Figure 5: A user intention record type to effect the movement of an object.

$$\left[\begin{array}{ll} & : \left[\begin{array}{ll} x & : e \\ p_{=basket(x)} & : t \\ p1_{=big(x)} & : t \end{array} \right] \\ r1 & \\ k1_{=1} & : \mathbb{N} \\ x2_{=\iota(r1,k1)} & : e \\ & : \left[\begin{array}{ll} x & : e \\ p_{=apple(x)} & : t \\ p1_{=red(x)} & : t \end{array} \right] \\ r & \\ k_{=1} & : \mathbb{N} \\ x1_{=\iota(r,k)} & : e \\ ev1_{=INTO} & : es \\ x_{=addressee} & : e \\ ev_{=PUT} & : es \\ p3_{=obj(ev1,x2)} & : t \\ p2_{=indObj(ev,ev1)} & : t \\ p1_{=obj(ev,x1)} & : t \\ p_{=subj(ev,x)} & : t \end{array} \right]$$

Figure 6: A DyLan parse record type.

and max functions in (14) and (15), respectively.

$$human.intention = \arg\max_{i \in I} p(wb : i | wb : e) \quad (14)$$

$$Ev(human.intention) = \max_{i \in I} p(wb : i | wb : e) \quad (15)$$

In our current implementation, e simply consists in judgements on the $human.c{-}utt.parse$ and $objects$ fields of wb, but it can be more than these, and in future, we plan to learn which parts are relevant for estimating user intentions.

In our current implementation, to calculate the conditional likelihood $p(wb : i \,|\, wb : e)$ for two given RTs i and e, we create a directed graph of the current parse RT based on its field dependencies, beginning from the head event field $e_{=PUT}$ (which determines the action), and recursively traverse all fields which depend on it, applying the relevant type classifiers. We match the field values in the embedded entity restrictor RTs (labelled r, $r1$ etc. within the parse record types like (6)) such as $apple(x)$, to the low-level classifier results encoded in the $objects$ field of wb. If the relevant type judgement (e.g. $apple(x)$) appears in the parse, the corresponding low-level classification (e.g. $c_{apple(x)}$) for each object will be used. An example of the probability judgement of obj_9 being classified as type $apple$ with probability 0.75 whilst grounding that object as the sole object in the set $intention.objects$ of a candidate intention is as follows:

$$p(wb : \left[\begin{array}{ll} parse & : \left[r : \left[\begin{array}{ll} x & : e \\ p_{=apple(x)} & : t \end{array} \right] \right] \\ intention & : \left[\begin{array}{ll} objects & : \{obj\_9\} \\ g & : G(objects, parse.r) \end{array} \right] \end{array} \right]) = 0.75 \quad (16)$$

When multiple classifiers are applied to entities, the product rule is used to multiply the probability of the relevant fields for a given object, assuming independence as described. The overall likelihood of $wb : i$ is calculated recursively, beginning with the likelihood of the embedded RTs such as $intention.goal$ and the target objects $intention.objects$. The likelihood of the judgements of each of the embedded fields is multiplied together to get the overall probability of the intention, as in Fig. 7 for combing the red and apple classifier judgements to obj_9.

This grounding process described for atomic type judgements is applied throughout the intention classification steps, where, typically for the example parse in (6), if $x2$ is resolved to obj_2 as shown in Fig. 1 and $x1$ is resolved to obj_9, then, adding the grounding predicates, the final $intention$ field of wb would be as follows:

$$\left[intention : \left[\begin{array}{ll} goal & : \left[\begin{array}{ll} lm & : \{obj\_2\} \\ rel\_loc & : INTO \end{array} \right] \\ g1 & : G(goal.lm, parse.x2) \\ objects & : \{obj\_9\} \\ g & : G(objects, parse.x1) \\ action & : PUT \end{array} \right] \right] \quad (17)$$

4.2.1 Quantification classifiers and cardinality of sets

As we showed in Section 3.2, for type judgements involving sets (set types), the probability of a type judgement that a certain field's value has a certain set of members, in general the probability is equivalent to the product of each member of the set being a member of it, as in (11). However, we assume all expressions involving objects in this domain are quantified, even if implicitly. We provide three different *quantification* classifiers for definite/unique quantification, existential quantification and universal quantification. For each of

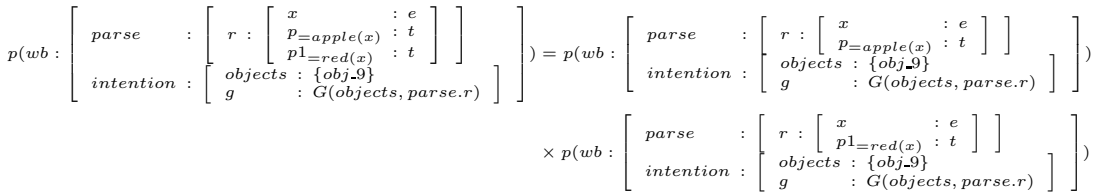


Figure 7: Combining probabilities for independent extensional classifiers to compute the probability of a given restrictor record type referring to a given object.

these we include cardinality of the object set as part of the classification process.

In (18) we define the ι-quantification function classifier for definite noun phrases in instructions like 'pass the three apples' where k=3 or for non-plural references such as in 'pass the apple' we implicitly assume k=1. The function simply takes a domain of type judgement on a set of objects which is grounded, then overrides that grounding to an ι predicate which specifies the cardinality k. 1 is returned if the cardinality of the set is k, else 0 is returned.

$$p((\lambda x : \begin{bmatrix} s : set \\ a : Type \\ g : G(a,s) \end{bmatrix} . \; x : x \boxed{\mathbb{A}} \begin{bmatrix} k : \mathbb{N} \\ x : \iota(a,k) \\ g : G(x,s) \end{bmatrix})(r)) = \begin{cases} 1 & \text{if } |r.s|{=}k \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

When we combine the application of this simple function classifier with the probabilistic type judgements themselves we get (19).

$$p(r : \begin{bmatrix} s : set \\ a : Type \\ k : \mathbb{N} \\ x : \iota(a,k) \\ g : G(x,s) \end{bmatrix}) = \begin{cases} 0 & \text{if } |s|{!=}k \\ \prod_{obj \in s} p(obj : a) & \text{otherwise} \end{cases} \quad (19)$$

This new classifier calculates the probability a given definite numerically quantified expression refers to a given object set s given the parse and a size k. 0 is returned if the cardinality of the set is not k, else it returns the product of the probabilities for each object in the set s being of the restrictor record type a.

For existential ϵ-quantification, we formulate a classifier in (20) for the indefinite noun phrases within such instructions as 'pass (any) three apples' where k=3 or 'pass an apple' where as we did for the ι classifier we assume k=1 and for 'pass some apples' we assume implicitly that $k \geq 2$. The classifier calculates the probability of an existentially (ϵ) quantified expression from a parse referring to a given object set s which has cardinality k. Again, 0 is returned if the cardinality of the set is not k, but the difference to the ι classifier is that 0 is returned if a, the restrictor record type, is judged to have a probability of referring to some $obj \in s$ of under θ, a confidence threshold determined experimentally. If both these conditions are not fulfilled, then it returns the product of the probabilities for each object in the set s being of the restrictor record type a. This formulation with the θ threshold allows the robot to question whether there is an example of the restrictor type judgement in the scene of the user. In future, we would like to experiment with active learning by adjusting θ if no suitable set of objects can be found.

$$p(r : \begin{bmatrix} s : set \\ a : Type \\ k : \mathbb{N} \\ x : \epsilon(a,k) \\ g : G(x,s) \end{bmatrix}) = \begin{cases} 0 & \text{if } |s|{!=}k \\ 0 & \text{if } \exists obj \in s.p(obj : a) < \theta \\ \prod_{obj \in s} p(obj : a) & \text{otherwise} \end{cases} \quad (20)$$

Finally, in (21) we formulate a universal τ-quantification classifier for noun phrases such as that in 'pass all the apples', where the classifier calculates the probability of a universally (τ) quantified expression from a parse referring to a given object set s. There is no cardinality requirement, however, like the ϵ classifier, 0 is returned if there is an object obj in s for which $p(obj : a)$ is under θ, a confidence threshold determined experimentally, else it returns the product of the probabilities that a refers to each object in the set.

$$p(r : \begin{bmatrix} s : set \\ a : Type \\ x : \tau(a) \\ g : G(x,s) \end{bmatrix}) = \begin{cases} 0 & \text{if } \exists obj \in s.p(obj : a) < \theta \\ \prod_{obj \in s} p(obj : a) & \text{otherwise} \end{cases} \quad (21)$$

5 Example application in a real system

In Fig. 8 we show the entire computation graph for computing the probability of the world belief being of the top record type, including the parse for

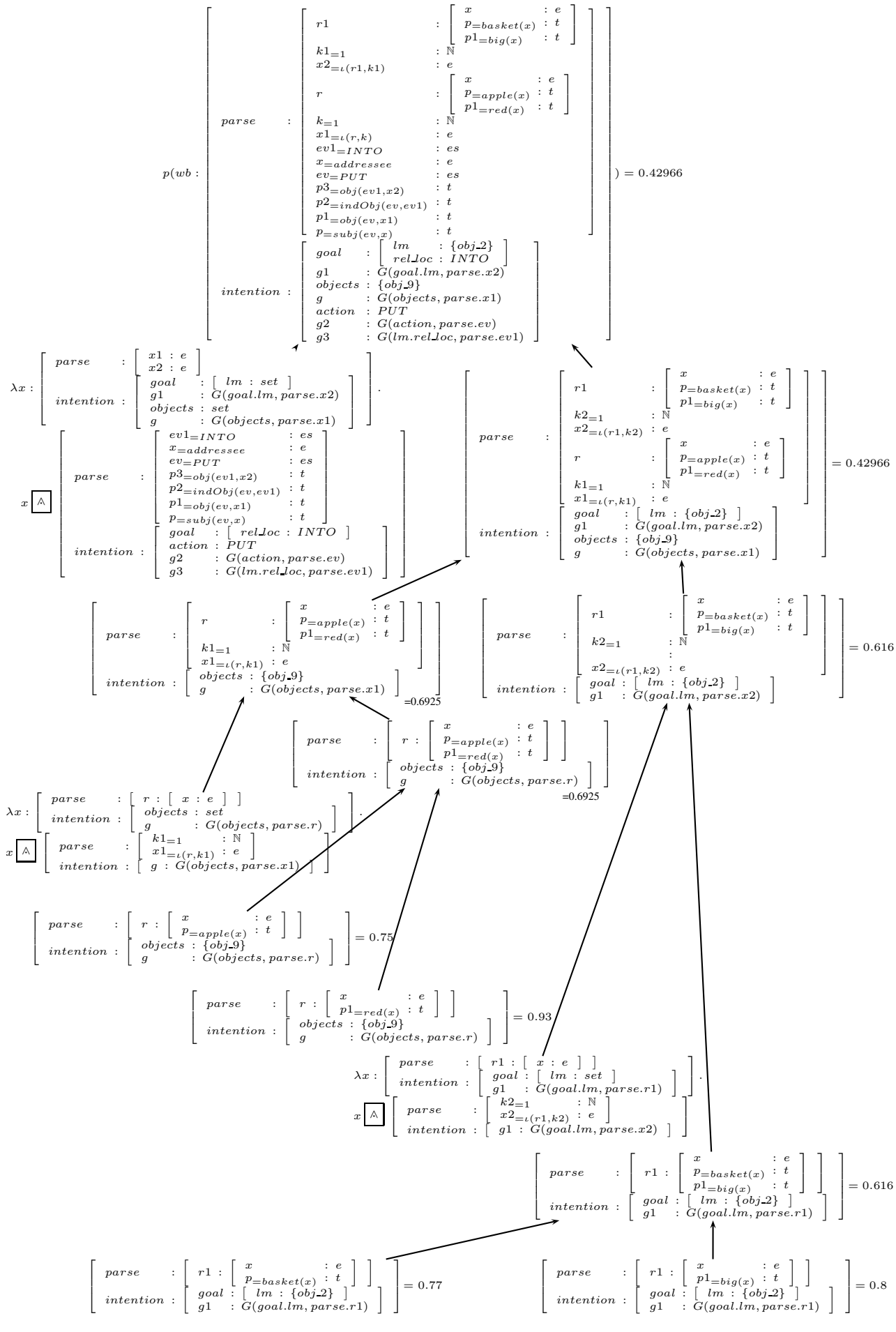
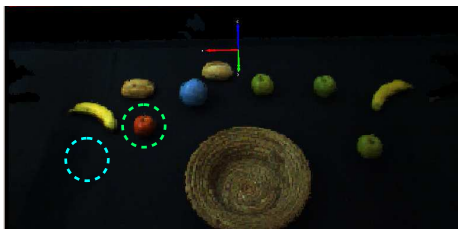
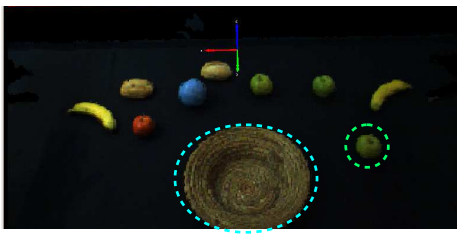


Figure 8: Graph for computing the probability of the world belief being of the top record type given parse for "Put the red apple in the big basket". Atomic child node probabilities are multiplied together. Lambda functions (left child) application probabilities are conditional on argument nodes (right child).



put the apple in front of the banana



... in the basket

Figure 9: Syntactic ambiguity causing the system changing its top hypothesis about the user's intention.

the utterance "Put the red apple in the big basket". Here we show the conditional probability $p(wb : i \mid wb : e)$ where i is the intention in Fig. 5 and e simply consists of the $human.c{-}utt.parse$ and $objects$ fields of wb. The candidate type judgement is first decomposed into its different type judgements from the top-down in the way shown before the probabilities are calculated. We only include the relevant low-level extensional classifier probability outputs rather than the raw features at the bottom nodes of the graph. The probabilities are calculated bottom up. One can see for non functional type judgements, the child node probabilities are multiplied together, as was shown in Fig. 7.

The two ι function classifier applications operate simply as in (18), outputting 1, as the cardinality of the sets of the $objects$ and $goal.lm$ fields of the $intention$ frame matched the values shown in the parse, both being 1.

For the function application involving the relation $INTO$, the probability of application to its argument record type behaves like a conditional node in a Bayesian network behaves with regards to its differing possible input values, effecting a conditional probability function or table as explained in Section 2.4. In this particular case, the function takes as a domain the two e-type fields in the parse $x1$ and $x2$ grounded into the $intention$ such that they are grounded references to the objects in the $objects$ and $goal.lm$ fields respectively. This function maps that domain to the part of the parse containing the $ev_{=PUT} : es$ field being grounded into the actual action PUT and the $ev1_{=INTO} : es$ field being grounded into the $goal.rel\_loc : INTO$ judgement of the intention. We formulate this as a simple classifier which returns 1 if the application is possible, based on the position and size properties of the objects, and 0 otherwise. Here obj_2 is judged to be a legitimate landmark for obj_9 to be placed into, so the resulting conditional probability of $goal.rel\_loc : INTO$ is 1. It is possible to turn these into fully fledged real-valued conditional probability functions, but we only present their potential for complex functions and leave this for future work.

5.1 Processing ambiguous instructions

The example showed how the system applies to a single parse and a single candidate intention which in this case is the most likely one for the parse and the world belief. In practice, the system is continuously maintaining a disjunction of probabilistic record type judgements, including for a beam of the top parses from the DyLan parser.

Given that the parsing hypothesis and the intention classification interact, our system in fact allows the different processes to help each other. For example the online disambiguation of parsing attachment ambiguity such as that in Fig. 9, where the first 'in front of the banana' is taken to be a goal location argument and not a modifier to 'the apple' because the parse is the most likely, but this decision is reversed once the user continues talking as 'in the basket' is then taken to be a goal location argument and the original most likely parse is removed from the top spot.

6 Conclusion

We have given an overview of a types-as-classifiers approach to dialogue processing in human-robot interaction. We believe our approach is complementary to the words-as-classifiers approach to reference resolution (Kennington and Schlangen, 2015), and we believe it brings several advantages. Firstly, it is not constrained by individual word classifiers alone, but can use the structure from a parser to compute likelihood of complex intentions, all the while maintaining word-by-word incrementality.

Secondly, it gives a uniform way to process different multimodal information such as robotic task and action states and visual and physical properties of objects within a dialogue state.

In terms of the general advantages over other machine learning systems, we claim that we would rather have interpretable, decomposible classifiers than uninterpretable flat representations– our approach allows for greater modularity, domain transferability and human understanding of the processing involved.

Acknowledgments

We thank the three reviewers for their useful comments, to Arash Eshghi and Robin Cooper for invaluable input, and to editors Chris Howes, Simon Dobnik and Ellen Breitholtz for their enduring patience. This work was supported by the DFG Center of Excellence EXC 277, the DFG Transregional Research Centre CML, TRR-169. Hough and Jamone are partly sponsored by the Alan Turing Institute partnership grant 'Learning collaboration affordances for intuitive human-robot interaction'.

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433.

Robin Cooper. 2005. Records and record types in semantic theory. *Journal of Logic and Computation*, 15(2).

Robin Cooper, Simon Dobnik, Shalom Lappin, and Staffan Larsson. 2014. A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS)*, Gothenburg, Sweden. ACL.

Atabak Dehban, Lorenzo Jamone, Adam R Kampff, and José Santos-Victor. 2016. Denoising auto-encoders for learning of objects and tools affordances in continuous space. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 4866–4871. IEEE.

Simon Dobnik, Robin Cooper, and Staffan Larsson. 2012. Modelling language, action, and perception in type theory with records. In *International Workshop on Constraint Solving and Language Processing*, pages 70–91. Springer.

Arash Eshghi, Matthew Purver, and Julian Hough. 2011. DyLan: Parser for Dynamic Syntax. Technical Report EECSRR-11-05, School of Electronic Engineering and Computer Science, Queen Mary University of London.

James J Gibson. 2014. The theory of affordances (1979). In *The People, Place, and Space reader*, pages 56–60. Routledge New York, NY, USA, London.

Afonso Gonçalves, João Abrantes, Giovanni Saponaro, Lorenzo Jamone, and Alexandre Bernardino. 2014. Learning intermediate object affordances: Towards the development of a tool concept. In *Joint IEEE International Conferences on Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014*, pages 482–488. IEEE.

Julian Hough. 2015. *Modelling Incremental Self-Repair Processing in Dialogue*. Ph.D. thesis, Queen Mary University of London.

Julian Hough, Casey Kennington, David Schlangen, and Jonathan Ginzburg. 2015. Incremental semantics for dialogue processing: Requirements, and a comparison of two approaches. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 206–216.

Julian Hough and Matthew Purver. 2017. Probabilistic record type lattices for incremental reference processing. In *Modern Perspectives in Type-Theoretical Semantics*, pages 189–222. Springer, Berlin.

Lorenzo Jamone, Emre Ugur, Angelo Cangelosi, Luciano Fadiga, Alexandre Bernardino, Justus Piater, and José Santos-Victor. 2016. Affordances in psychology, neuroscience and robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*.

Casey Kennington and David Schlangen. 2015. Simple learning and compositional application of perceptually grounded word meanings for incremental reference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 292–301.

Staffan Larsson. 2015. Formal semantics for perceptual classification. *Journal of logic and computation*, 25(2):335–369.

Staffan Larsson. 2018. Grounding as a side-effect of grounding. *Topics in Cognitive Science*, 10(2):389–408.

Vivien Mast, Zoe Falomir, and Diedrich Wolter. 2016. Probabilistic reference and grounding with pragr for dialogues with robots. *Journal of Experimental & Theoretical Artificial Intelligence*, 28(5):889–911.

Matthew Purver, Arash Eshghi, and Julian Hough. 2011. Incremental semantic construction in a dialogue system. In *Proceedings of the 9th IWCS*, Oxford, UK.

Deb Roy. 2005. Grounding words in perception and action: computational insights. *Trends in Cognitive Sciences*, 9(8):389–396.

RS Sundaresh and Paul Hudak. 1991. A theory of incremental computation and its application. In *Proceedings of the 18th ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages*, pages 1–13. ACM.

Andre Ückermann, Christof Eibrechter, Robert Haschke, and Helge Ritter. 2014a. Real-time hierarchical scene segmentation and classification. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 225–231. IEEE.

André Ückermann, Christof Elbrechter, Robert Haschke, and Helge Ritter. 2014b. Hierarchical Scene Segmentation and Classification. In *Robots in Clutter Workshop at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*.

Yanchao Yu, Arash Eshghi, and Oliver Lemon. 2016. Training an adaptive dialogue policy for interactive learning of visually grounded word meanings. In *17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 339.