

DIPLECS Deliverable D7.2

(In situ strategy induction system)

Grant Agreement number: 215078

Project acronym: DIPLECS

Project title: Dynamic Interactive Perception-action LEarning in Cognitive Systems

Funding Scheme: Small or medium-scale focused research project

Date of latest version of Annex I against which the assessment will be made: 5/10/2007

Period covered: from 30/05/2009 to 30/11/2009

Responsible Beneficiary: David Windridge, Surrey University UK

Tel: +44 (0) 1483 68 6048

Fax: +44 (0) 1483 68 6031

E-mail: d.windridge@surrey.ac.uk

Address: CVSSP, University of Surrey, Guildford, GU2 7XH

Contents

1	Introduction	3
1.1	Objectives	3
1.2	Context of Deliverable D7.2 within WP7	4
2	Overview of System	4
2.1	Example Scenario: Spanish Turn	6
2.1.1	'Model Acquisition' verses 'Model Rectification'	6
3	Work Details: Part 1 - Data Sparsity Experiments	7
3.1	Experimental Format	8
3.1.1	Logic-based classifier with sparse data	9
3.1.2	Logic and Decision-Tree classifiers with sparse data	10
3.1.3	Top-down feature respecification with sparse data	12
4	Work Details: Part 2 - Specification of Interface	13
4.1	Low-level Feature Vector interface	14
4.2	Sub-logical Metrical Interface	17
4.3	WP4- WP7 Interface Topology	18
4.3.1	Buffer Structure: The A / B buffer interface	19
4.3.2	Buffer structure: temporal considerations	23
4.3.3	Buffer structure: multiple consistent world models	24
5	Work Details: Part 3 - Implementation of a Fuzzy-logic deductive system.	25
6	Summary	28

1 Introduction

1.1 Objectives

WP7 (System Supervision and Strategies) deals, as its primary mode, with the inference of driving intention, and with the formation of globally-consistent scene-description appropriate to the execution of these intentions (i.e. global consistency need only be resolved with respect to scene descriptors relating to intentionality, in an implementation of the perception-action principle). Both scene-description and intention are implicitly hierarchical, meaning that consistency can be determined on an intra-level basis. In this way, we aim to produce a compact, affordance-based model of scene and intention.

A major issue for this approach is the reconciliation of *a priori* models of intention and driving (i.e. the ECOM model and the Highway Code, respectively) with the input from visual detectors, gaze and control inputs. Because these *a priori* models are both implemented in first-order logic (cf D7.1 for how this PROLOG-based process was initiated), the system must thus be capable of reconciling abstract deduction with pre-symbolic input from WP4. It must hence address the issues of symbol grounding [1] and symbol tethering [5]. (Symbol tethering involves the deductive extension and comparison of low-level predication with actual scene measurements, so that the Herbrand universe of the deductive system need only be constrained at relatively sparse points). The logic system must thus be adaptive, as well as being capable of generating top-down input to the detectors to give confidence feedback.

The current deliverable concerns the last 6 months activity building on the work of deliverable D7.1, with the intention of rendering it suitable for application in the context of the full DIPLECS system (Deliverable D7.1 constructed a prototype system for bootstrapping formalized strategies and associated logical representations).

Three major issues, in particular, have been addressed in order to render the previous work applicable in the context of the system as a whole. These are:

1. The Issue of Data Sparsity.

Deliverable D7.1 employed fully ground-truthed input data, defined at all levels of the intentional and scene-description hierarchy. Real input data is likely to be a very small subset of this, so the logic system has been tested and extended in the adverse condition of extreme data sparseness.

2. Implementation of a Fuzzy-logic deductive system.

The ground-truthed input data used for Deliverable D7.1 was of a binarised variety. Real input data will employ associated confidence values, and the logic system has been

extended to accommodate to this.

3. Specification of Interface

The format for interfacing of detector input with the logical deduction system has been formally elaborated through the consideration of concrete scenarios, and the logical structure modified accordingly. Issues relating to compatibility with the ICE-interface have also been addressed.

Beyond these objectives, the ongoing refinement and extension of logical predicate/clause structure has continued.

1.2 Context of Deliverable D7.2 within WP7

The current deliverable combines from work of WP7.1 (Determination of human strategies and corresponding symbolic representations), WP7.2 (Deduction of Appropriate strategy for any given situation) and WP7.3 (Inference of Current Strategy). Of these, WP 7.1 and WP7.2 have been brought to completion, with WP7.3 due for completion in month 32.

Completion of WP 7.2 involved the implementation of the ECOM control hierarchy in logical terms (specifically the tracking and regulating intentions) in a manner consistent with the highway code. ARMINES, as well as providing the psychological model, continues to provide expert ground-truth annotation of data in terms of the ECOM categories, and interdisciplinary input.

WP 7.1 thus specified a series of intentional categories for classification and stochastic learning problems relating to driver intention, which were to be rendered logical consistent via the PROGOL-based system developed for WP 7.2. WP7.3 then proved the classification output. Since the logical reparamterisation method employed within COSPAL was not, in fact, determined to be useful within the DIPLECS context, an alternative mechanism for adaptive behavior was investigated; specifically, a top-down, global logical consistency-based feedback mechanism for redetermining detector confidence. This meant that the practical boundaries between WP 7.1, WP7.2 and WP7.3 were not as absolute as those specified in the Technical Annex.

Deliverable D7.2 constitutes the first fruits of the WP4-WP7 collaboration axis.

2 Overview of System

The following are the main intended capabilities of the deductive system (WP7SYS) when in implemented in situ.

1. ECOM State Annotation with respect to detector/control inputs (this is the primary WP7SYS output mode)

The output from this capability is the logically stabilized and contextualised result of the application of the ECOM state classifiers to the current driving scene (the system receives detector input at arbitrary levels of the visual hierarchy: eg we may detect a car and a T-junction without any corresponding detection of the intermediate junctions structure or lane-occupancies). The system then stochastically computes the ECOM hierarchy from these inputs combined with the driver's gaze, signal and control inputs. The deductive module determines self-consistency of the road configuration and ECOM intention with respect to to each other.

2. Top-down Feedback to WP4 (the secondary WP7SYS output mode)

This has two distinct modes:

a) Online Mode: this involves deductive instantiation of logical variables via resolution

The deductive system effectively 'fills in the blanks' of scene description passed from detectors. This will sometimes involve instantiation of symbol attribute labels eg attaching a new *sign_label* of unknown type to its functional specification. This information is fed back to WP4 for possible classifier respecification.

b) Offline Mode: this rectifies fundamental faults in low-level scene description eg faulty detectors/wrongly attached symbol/functional pairs (ie where serious errors are made with high confidence)

When variable instantiations don't give rise to a sufficiently large 'consistency set' of predicates, a 'global consistency check' can sometimes rectify the situation. This involves sequentially removing feature predicates (ie detector outputs) and recalculating the associated predication to determine whether this improves overall global consistency as measured by the final size of the consistency set. We can then 'weight' feature detectors on the basis of this consistency (currently a binary weighting; but, ultimately, a fuzzy threshold will be applied).

That is, we use global deductive logical consistent to make the best rectification assertion possible; ie we need to compute the smallest detector change that gives rise to the largest maximally-consistent set of deductions

3. The Warning System (this is an occasional WP7SYS output mode)

This triggers a warning flag on the basis of a significant disparity between the Observed Driver Output (ODO) and the Predicted Driver Output (PODO) at the predicate level (the

system in this mode does not distinguish between discrepancy caused by erroneous scene representation and erroneous driver activity; i.e. it assumes a perfect world model).

2.1 Example Scenario: Spanish Turn

The 'Spanish turn' scenario involves an incompatibility with the low-level visual feature detection and the high-level logical context modelling. In particular, the Spanish turn sign depicts a right-turn with a 'fly-over' configuration, so that immediately after the right-turn branches off, it appears to cross the original road and continue to the left. Thus, a situation is envisaged in which a low-level visual detector classifies the novel sign-type as being of the 'left-turn sign' class with a high degree of certainty on the basis of its appearance, while, at the same time, the context of the driver's actions (applying a right-turn indicator, etc), suggests that in fact the sign relates to a right-turn.

We indicate here how the in-situ top-down logical feedback has been implemented so as to be able, in principle, to adaptively resolve this problem:

2.1.1 'Model Acquisition' verses 'Model Rectification'

The 'Model Rectification' approach to the Spanish turn problem involves unlearning a previously learned model and would be implemented off-line as follows within in WP7SYS.

As driving progresses, the logic systems attempts to build the most consistent model of the scene at all levels of the representational hierarchy. This involves multiple hierarchical and temporal logical extensions of the inputs from LiU. Consequently, if the car comes to what is asserted with high confidence to be a right-turn sign, and the car in fact turns left, then the maximal logically-consistent set of predicates would rapidly tend to a very small cardinality. (This could perhaps in itself be used to generate a warning of generalized inconsistency; particularly if it were repeated.)

However, it becomes computationally complex if we are to try to establish exactly what was causing the inconsistency, given that there are any number of other detectors (eg 'lane-number', 'road-direction', 'junction-type', 'indicator') that may be at fault. To establish this, we progressively remove every detector from the scene description in the manner set out in D7.1, and determine whether the cardinality of the maximal logically-consistent set increases significantly. The problem of rectifying the wrong Spanish turn sign association at the logical level would then involve exploring alternative sign-concept mappings; being non-trivial, this is an off-line problem.

On the other hand, Spanish turn model *acquisition* is much more straightforward. Eg, we would have the following (highly simplified) predicate input:

steering_wheel_left_turn(DIPLECS_Car_at_Time_t).

left_indicator(DIPLECS_Car_at_Time_t).
sign_object_detected(label_1, t).
sign_type(label_1, X).

and clauses:

$\forall y \exists x \text{ sign_type}(Y, X).$
 $\forall y \forall t \text{ sign_type}(Y, \text{Left_Turn}) \ \& \ \text{sign_object_detected}(Y, t) \Rightarrow \text{Left_junction_Traverse}(\text{intention}, t)$
 $\forall t \text{ Left_junction_Traverse}(\text{intention}, t) \Rightarrow \text{steering_wheel_left_turn}(DIPLECS_Car_at_Time_t).$

That is, the detector inputs indicate that a high saliency object in a left turn scenario has been classed as a sign, but of unknown type. That is, the system 'knows' a priori that the problem is one of sign/function mapping.

This would lead to the straightforward deductive resolution:

sign_type(label_1, Left_turn_sign).

Thus, the ordinary online deductive instantiation mode of WP7SYS naturally solves the problem.

There is some potential for reducing the complexity of the offline problem by incorporating prior knowledge of the intention-object mapping, so that the nesting of intentions potentially makes the search space smaller at the logical level. However, this is still likely to remain an offline problem solving mode, given the remaining complexity. In essence, the system still needs to assess global consistency to make the best rectification assertion possible, so that we need to compute the smallest detector change that gives rise to the maximally consistent set of deductions (parsimony principle).

3 Work Details: Part 1 - Data Sparsity Experiments

Previous experiments on ground-truthed data to determine the feasibility of top-down feedback to WP4 (reported in D7.1) were redesigned for testing the effectiveness of this mechanism in a realistic environment. The predominant difference between this environment and the previous experiments is in terms of the sparsity of the input (previously, we assumed the availability of detectors across the full scene-description hierarchy, so that, for example, distinct road and junction-type detectors existed, irrespective of the fact that junction type is logically derivable from road topology).

Realistic sparse input consist in the following detectors: road detectors, external car detectors (with indicator/brake signals), sign detectors (if not sign-type identifications), traffic light and light states detectors, and finally the intersections of gaze with bounding

boxes around these entities. We are thus using 33 features only (instead of 772 as previously used for predicates), and thus all lane-occupancy information has been completely removed, and so multiple consistent car-lane allocations are possible with the existing predicates.

3.1 Experimental Format

A range of inner-urban junction scenarios within Dataset 8 were selected for ground-truth annotation on the basis of their provision of good-quality LIDAR data for junction topology propagation (cf D7.1), and exemplification of the Regulating and Monitoring intentions of the ECOM control model, along with their conditional logic dependencies on environmental entities such as traffic lights and signs.

In particular, 6 cross-road traversing scenarios are considered, consisting of 2 cases each of left-turning, right-turning, and 'straight-over' junction traverses. This set is a nearly maximally complex representation of the driving situation, in the sense that it is selected to effectively exhaust to all possible 'configuration-changing' driving scenarios: that is, all other driving situations (lane changes, roundabout-traverses, etc) can be considered degenerate instances of these cases.

Note that individual levels of the ECOM hierarchy, l , are considered to be mutually temporally exclusive; individual levels, however, may be simultaneously operative. (More generalized levels at the top of the hierarchy will tend to be operative over a longer period of time than sub-level intentions). The intentional classification problem is thus one of simultaneous categorization of the unique item i^l applicable within each level (with the inclusion of a null item if necessary). That is, for the totality of individual frames we need to solve the mapping problem:

$$\forall l, X \rightarrow i^l : l = \underset{n}{\operatorname{argmax}} \{p(i^n|X)\} \quad (1)$$

(X being the feature vector).

We thus generate a full hierarchy of binary features so that, for instance gaze attention directed at a particular lane also includes the road that contains it, and the junction that contains the road, etc. We use a decision-tree algorithm for classification of the ECOM states on the basis of it's readily-interpretable results. In particular, it has the characteristic of defining a decision rule that is directly translatable into first-order logic. (We also tried CN2 in this context, but found that it did not provide better performance than decision trees). The output from the decision trees thus provides a discriminatively-stabilised per-frame ECOM-state input into the logic system, such that the logic system can test for global logical consistency in both spatial and temporal terms.

Thus the deductive logic system extends the intentional detection system to accommodate rule-like behaviors relating to Highway-code based intentional configuration changes that cannot be fully captured by stochastic correlation.

When placed in situ, the activity of the logical infrastructure within the strategy/supervision system is threefold (see earlier); however, in the most typical mode of operation, the system will construct a logically consistent world-model from the computer-vision system's input and use the conditional dependencies from the driver's gaze, signal and control inputs to determine the operating intention and sub-intention of the ECOM model at any given time.

3.1.1 Logic-based classifier with sparse data

It is also possible, however, for the logic system to deduce active ECOM states on a per frame of itself. As a performance baseline, the accuracy of these outputs are as indicated in figures 1 to 6 for the sparse input data set.

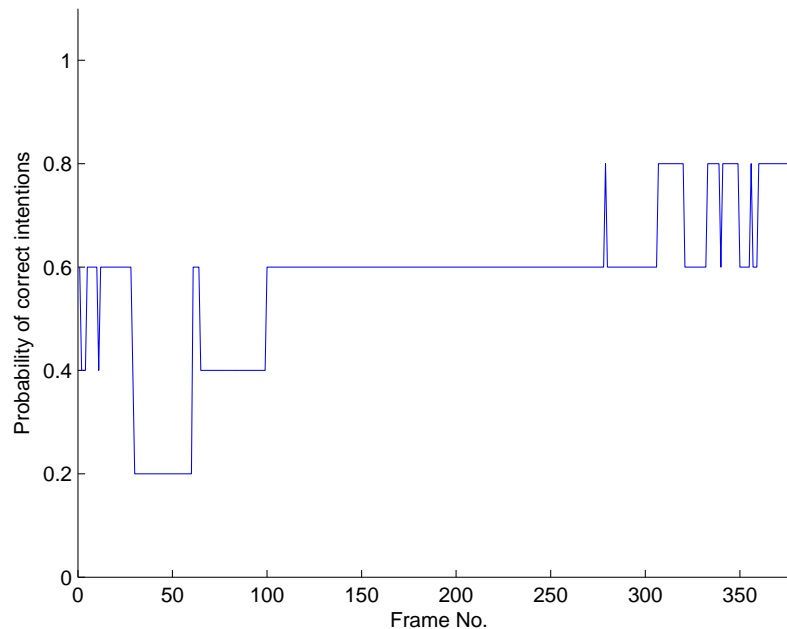


Figure 1: 1st Left-turn Scenario

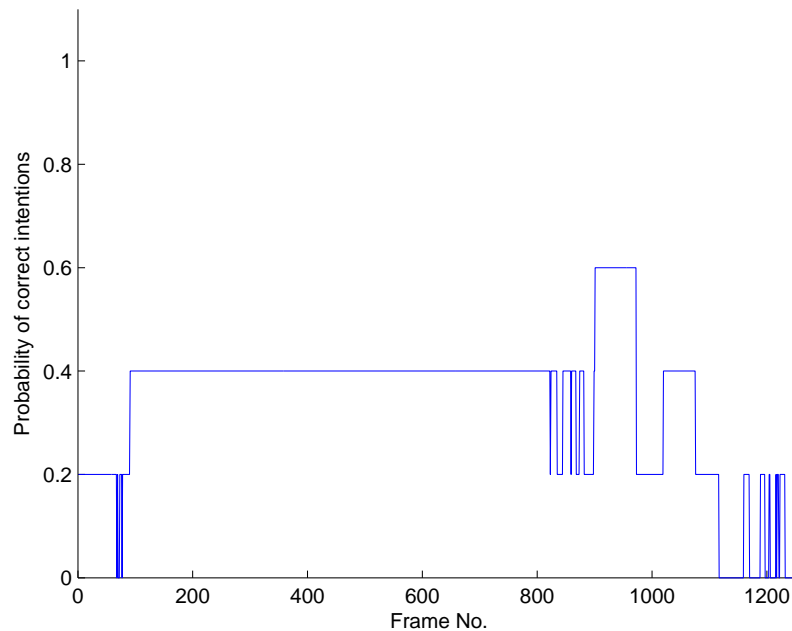


Figure 2: 2nd Left-turn Scenario

By comparison, the corresponding accuracy values for the complete (i.e. non-sparse) data set are given in figures 7 to 12 for the six junction traverse scenarios. As can be seen, very little performance compromise is involved in the transition to sparse data.

Superimposing per-frame decision tree accuracies on the above graphs, we see how the lack of temporal context is seen to detrimentally affect the per-frame accuracy of the decision trees (cf fig 13). (Decision tree accuracy does not vary as more temporal context is gained, unlike the purely logical classifiers)

The corresponding graph for the non-sparse data is given in fig 14.

3.1.2 Logic and Decision-Tree classifiers with sparse data

Since the errors made by the logic and decision-tree systems are uncorrelated, it is thus possible to combine them advantageously.

Decision-tree outputs are hence combined with logical deduction through their incorporation into the consistency testing and aggregation procedure. Under purely logic-based consistency testing it is, for instance, possible that error-containing sets are consistent with

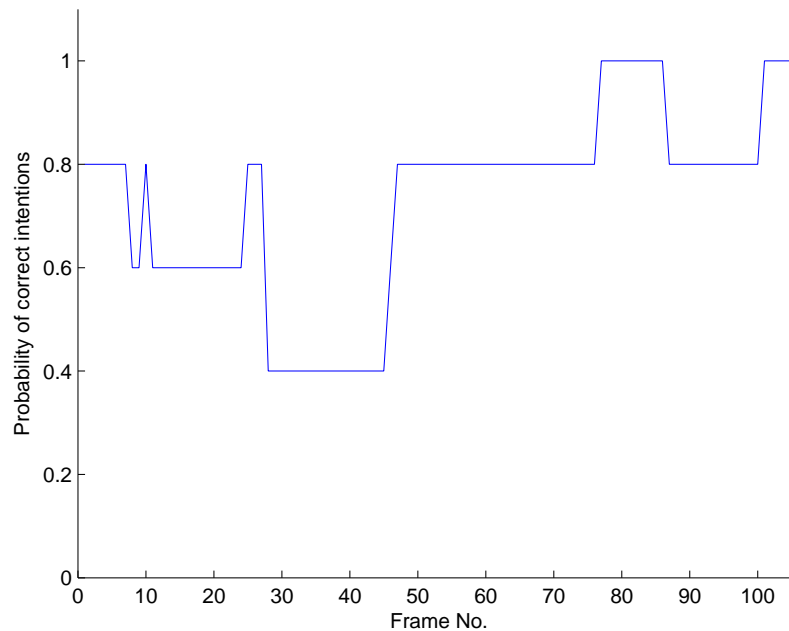


Figure 3: 1st Right-turn Scenario

each other by chance, leading to a false aggregation of consistent frames, and irrecoverable system error. We hence use the decision tree outputs to set thresholds on acceptance for the consistency set by requiring the agreement of the decision-tree outputs with that of the logic in order for the frame to be added to the consistency set. Otherwise the logic output is given as output by default, but not added to the consistency set. When a fully-inconsistent frame is detected by the logic system, the output is switched instead to that of the decision tree.

With this decision-tree-based temporal-node-weighting, the accuracy of the composite system is very significantly greater than that of the individual systems, as depicted in fig 15 for the right-hand turn data set. For the non-sparse data, the results are as depicted in 16.

We have thus created an effective in situ system for composition of discriminative and generative classifiers of intentional behavior, that utilized stochastic and structural techniques to combine global and local context information. In particular, there appears to be no significant loss associated with the move to sparse features by virtue of the deductive extension of the scene-description and ECOM-intention hierarchies.

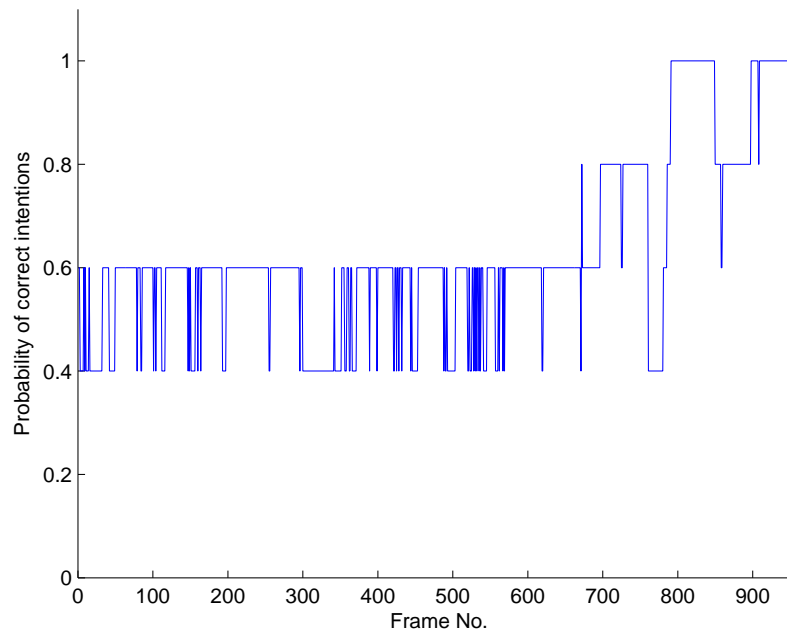


Figure 4: 2nd Right-turn Scenario

3.1.3 Top-down feature respecification with sparse data

We simulate the effects of noise on an individual feature by randomly replacing binary values for this feature within the frame vector by arbitrary binary digits. We do this according to a uniform random distribution with an average of failure probability of 0.2 per frame. The feature selected is no.19 (DIPLECS car position).

The aim is then to determine which of the 33 (of 772 possible) features is subject to this additive noise purely on the basis of global logical consistency.

This is done by sequentially removing feature predicates (ie detector outputs) and recalculating the associated predication of each frame for all of the features with the sparse data set, and determining whether this improves overall global consistency as measured by the final size of the consistency set. It is then possible to reweight feature detector confidence on the basis of this consistency. This process forms the basis of the logic system's feedback to WP4, and is implemented via the Buffer A / Buffer B interface (see below).

The results of the application of this method are shown in fig 18 for sparse features. We see that, as for the non sparse data (fig 18), a clear peak in the magnitude of the consistency

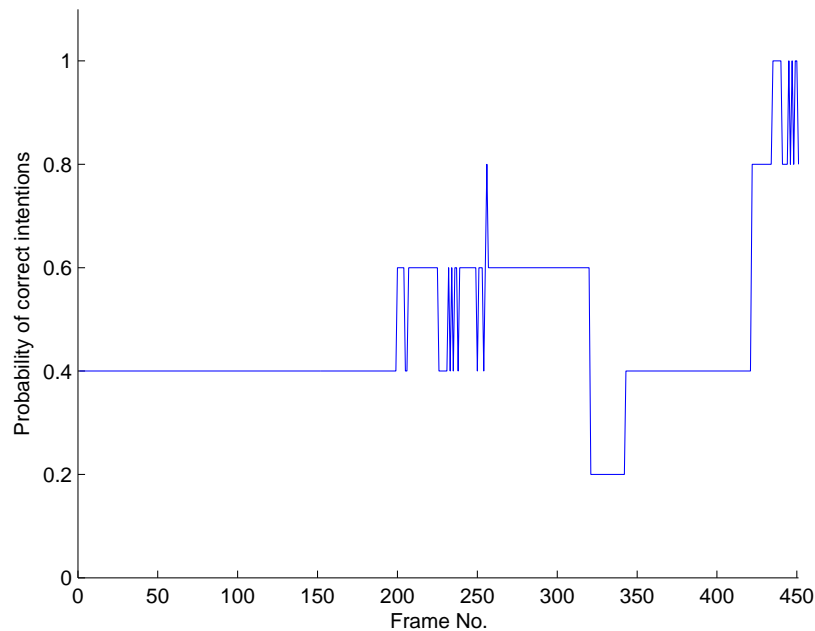


Figure 5: 1st Straight-over Scenario

set exists for the error-compromised feature. Sparsity is in fact advantageous here, in that it significantly reduces computation time.

It is critical that this measure of global consistency correlates with the accurate prediction of the ground truth values. We see from fig 19 (giving the average accuracy over the whole time-sequence) that this is indeed the case. The existing result for non-sparse data is as given in fig 20.

Blanking the features associated with the peak in global consistency therefore acts to increase overall system accuracy in the manner intended.

4 Work Details: Part 2 - Specification of Interface

In order to meet the requirement of implementing the WP7SYS prototype in an ICE-compatible manner, it was necessary to modify existing components of the system, and to extend the specification of the prototype system in three distinct aspects: modified low-level feature vector interfacing, the introduction of a sub-logical metrical interface to the

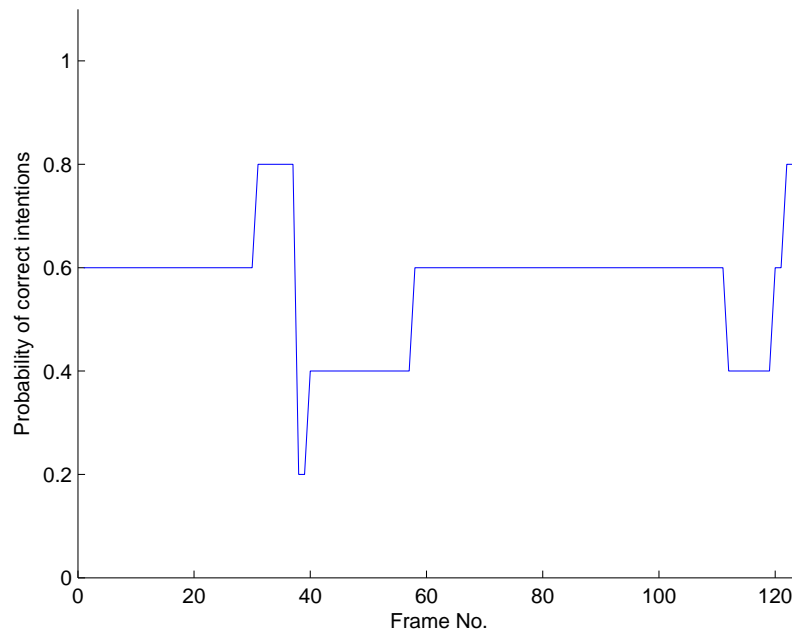


Figure 6: 2nd Straight-over Scenario

existing predication, and the designation and implementation of the WP4-WP7 Interface Topology. We thus proceeded as follows:

4.1 Low-level Feature Vector interface

The raw input (currently from the ground-truthing), consists of the following data classes generated for each frame frame:

1. Gaze bounding-box occupancies (ie a binary vector detailing whether the gaze is within a bounding-box around key road entities: lanes, cars, signs, lights etc)
2. The road topology (current junction configuration in order from front ie eg *left – hand – junction*, *right – hand – junction*, *X – road*, end-road-T-junction , etc [generally only 1 junction ahead is visible])
3. The set of visible car labels: *car_label_1*, *car_label_2*, etc (cars are given a new token identifier as they are detected, and for their duration in sight - if visual continuity is lost

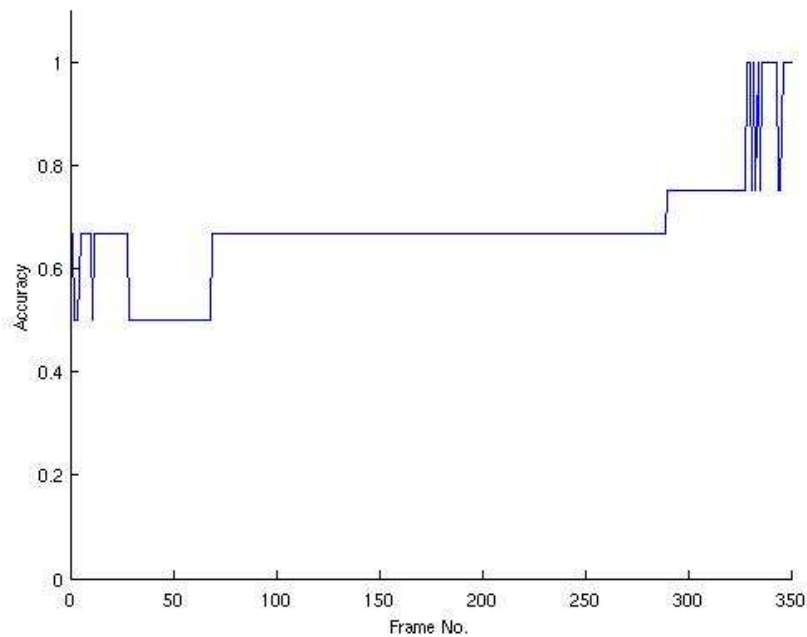


Figure 7: 1st Left-turn Scenario (non-sparse data)

the same car can have multiple labels [ie its for the logic to decide if they are the same entity on the basis of global consistency, not the detectors])

4. Car positions (ie best estimate of lane occupied)

5. Car ordering (ie lane order of cars from driver's position: ie [*lane_1_driver_side:car_label_1, car_label_5*], [*lane_2_driver_side: car_label_2, car_label_3*], etc

6. coarse-grained driver-relative car directions (driver-wards, leftwards, right-wards, anti-driver-wards)

7. Visible car signals if any (brake, left/right indicator) ie [*car_label_1: left*], [*car_label_2: brake*], etc

8. The set of visible signs with road positions (usually driver's right hand side [DRHS])

9. The set of visible traffic lights with positions [*light_1: DRHS*] and states (note as well as [red, amber, red+amber, green], there is as a null state [null] between light changes)

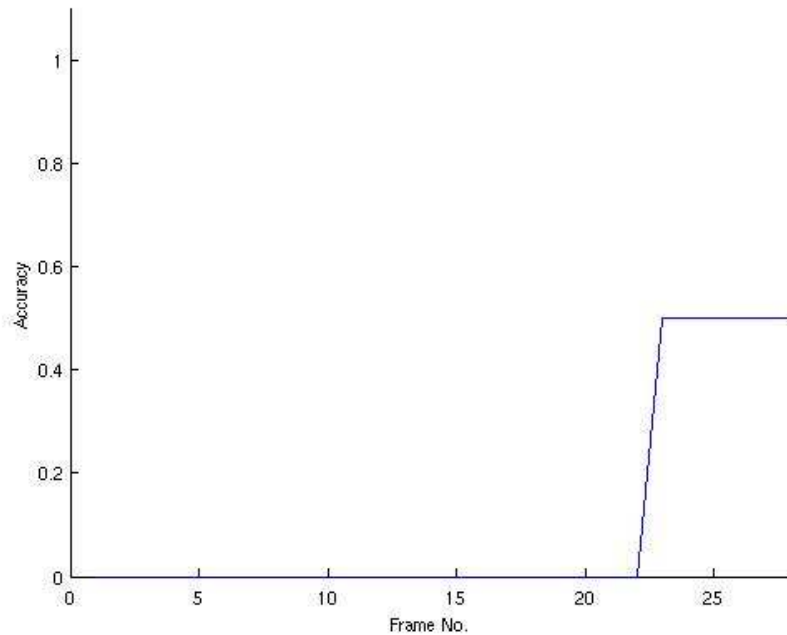


Figure 8: 2nd Left-turn Scenario (non-sparse data)

10. The set of detected pedestrian crossings and road positions (positions being: *driver's_road*, *left – hand_road*, *right – hand_road*, and also *opposite_road* when at a cross-road)

11. coarse-grained pedestrian positions/directions if close to crossings

The second stage of feature-vector then consists in the automatic generation of the full per-frame binary vector, which includes temporal states and an exhaustive set of interrelations between objects (constituting a 722-dimensional vector for each frame).

This process was originally accomplished in MATLAB for D7.1, but has now been incorporated within the logic module for continuity of mechanism, and compatibility with the ICE wrapper.

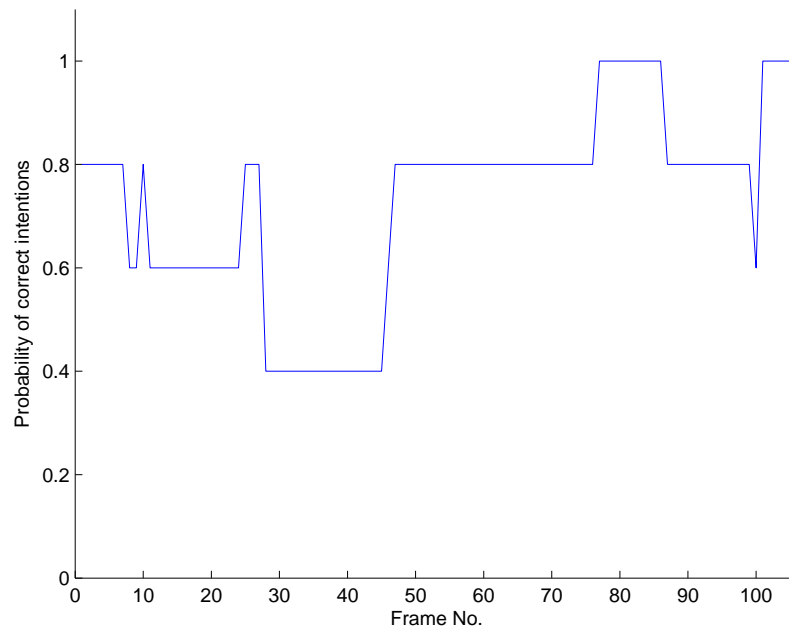


Figure 9: 1st Right-turn Scenario (non-sparse data)

4.2 Sub-logical Metrical Interface

We have also established the necessity for intermediate logical structure to perform internal conversions from screen position & depth with respect to nominal junction outline in order to map onto the predicates (eg *Car_lane_number*, etc) referred to in D7.1.

This will be constructed once the ICE interface is in place (the ICE interfacing is thus an ongoing process). We will hence be provided by WP4 with information about the coarse horizontal position of other cars in the image and a coarse size (= inverse distance), which should be sufficient to approximate relative positioning of all entities in the junction (the latter information obtained from GPS data). We are therefore building a set of intermediate structures to 'predicativise' this information.

Thus, rather than use a projected model of the road-outline, we can utilize prior information about road width and road intersection angle (from the GPS coordinates) to substitute for it. This means that we will extend structures below the symbolic level into the metric level to build intermediate representation.

Thus, we aim to be able to use any reasonable indication of junction positions relative

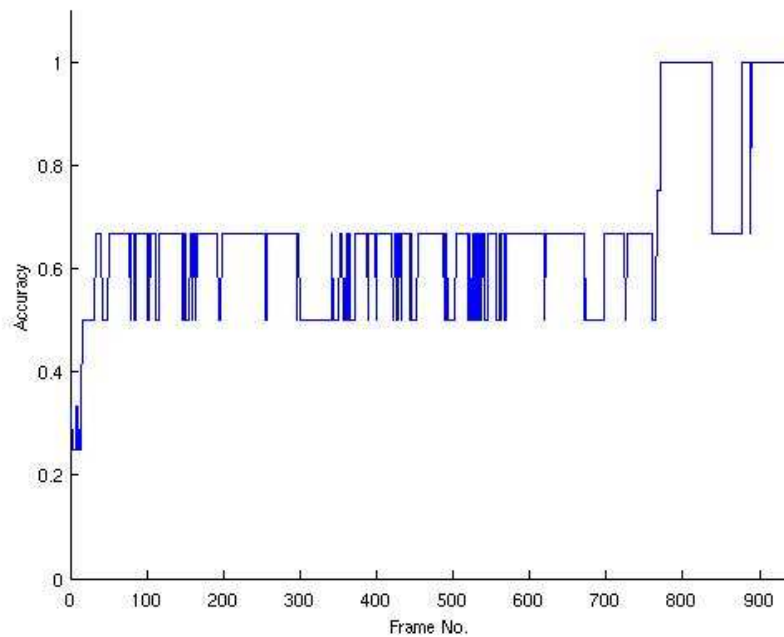


Figure 10: 2nd Right-turn Scenario (non-sparse data)

to detected (eg if we detect a potential left-hand junction at screen height X_1 , and a car in the left field of view at screen height $X_1 + \delta X$, then the proximity-based association of the of these two entities should be indicated by the system). The logic system would then deduce that the DIPLECS car is in the vicinity of a left junction from the GPS, and that any car with an extreme left screen displacement is likely to be on this road.

4.3 WP4- WP7 Interface Topology

WP7 mediates between the ECOM annotations and the detector input from LiU. In bottom-up terms, WP7SYS hence take input from LiU in the form of concrete (predicatisable) scene detections, along with any associated confidences, and use the annotation provided by ARMINES to learn the associated ECOM labels. As well as stochastic learning of the mapping, this involves long-range deductive logical consistency considerations to compute the entire ECOM hierarchy. Moreover, in top-down terms, we have the potential to take these global consistency considerations into account in order to provide concrete feedback to WP4. At the moment this takes the form of a 'detector error' flag (see D7.1). We also

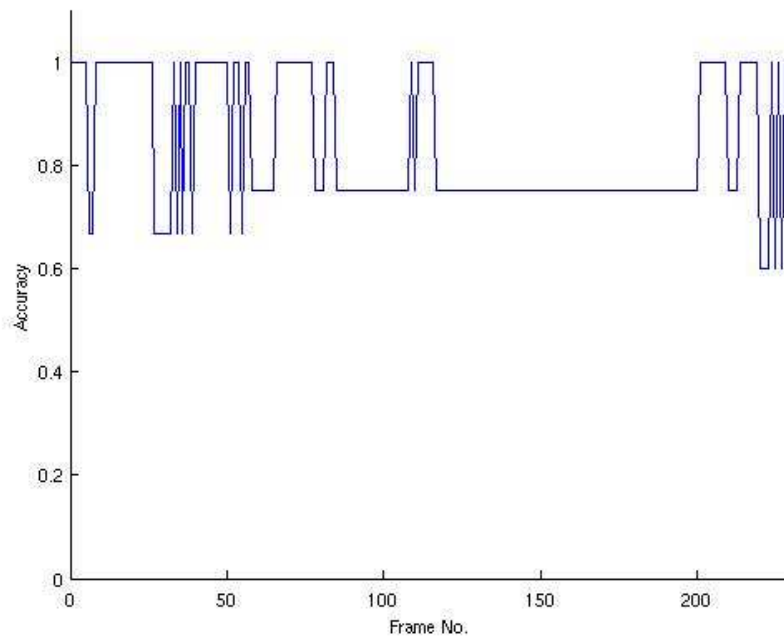


Figure 11: 1st Straight-over Scenario (non-sparse data)

anticipate that this will involve more detailed instantiation-based data.

It is in this sense that adaptivity to atypical situations occurs within WP7SYS; we do not change *a priori* logical structures, but only the mapping of the logic onto the scene description labels.

4.3.1 Buffer Structure: The A / B buffer interface

We hence introduce a 'buffer' system to intermediate between the logical predicate and WP4 symbol system (and which extends the scope of WP7 to the metrical level). This buffer structure is integrated via the ICE interface.

WP7SYS thus takes input from buffer A, and the logic system tries to fill in the 'blanks' for unspecified entities within the buffer using overall consistency check on the basis of prior inputs (along with their confidence values). A "main" function calls appropriate subfunctions and writes outputs to a buffer B. Structurally, the relation of Buffer A to Buffer B is as depicted in fig 21

Thus, Buffer A is the general WP4 write buffer, and Buffer B is the general WP7

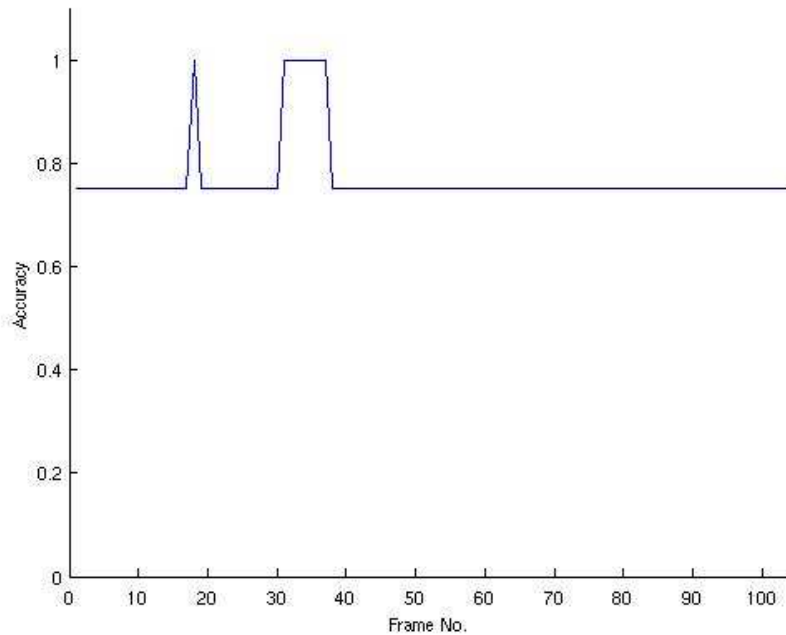


Figure 12: 2nd Straight-over Scenario

write buffer. Because WP7 deals with logical consistency, and WP4 deals with detector stabilization, Buffer A is sparse while Buffer B is more complete (and possibly multiply-complete for several mutually-consistent complete world-models).

We illustrate this in the case where multiple tracks are resolved into a single car entity. Thus, a car-track with *label_1* at time *t_1* and position *p_1* might be written to buffer A as *detected_car_track(label_1, t_1, p_1)*. A second car-track would be written as a separate predicate assertion: *detected_car_track(label_2, t_2, p_2)*. Within the logic buffer (buffer B) there is an entity defined by the predicate *logic_car(label_a)* triggered by the first track detected:

$$\text{if } \text{detected_car_track}(L, -, -) \ \& \ L = \text{label_1} \Rightarrow \text{possible_position}(\text{label_a}, \text{present}, p_1) = \text{.true.}$$

However there are also clauses that say (very approximately):

$$\forall P \text{ path_continuation}(P, P_old) \ \& \ \text{possible_position}(L, \text{present}, P_old) \Rightarrow \text{possible_position}(L, \text{past}, p_1) \ \& \ \text{detected_car_track}(L1, -, P) \ \& \ L1 = \text{label_1} \Rightarrow \text{possible_position}(\text{label_a}, \text{present}, P) = \text{.true.}$$

which would have the effect of potentially allocating both the track predicates - *detected_car_track(label_1, t_1, p_1)* and *detected_car_track(label_2, t_2, p_2)*.

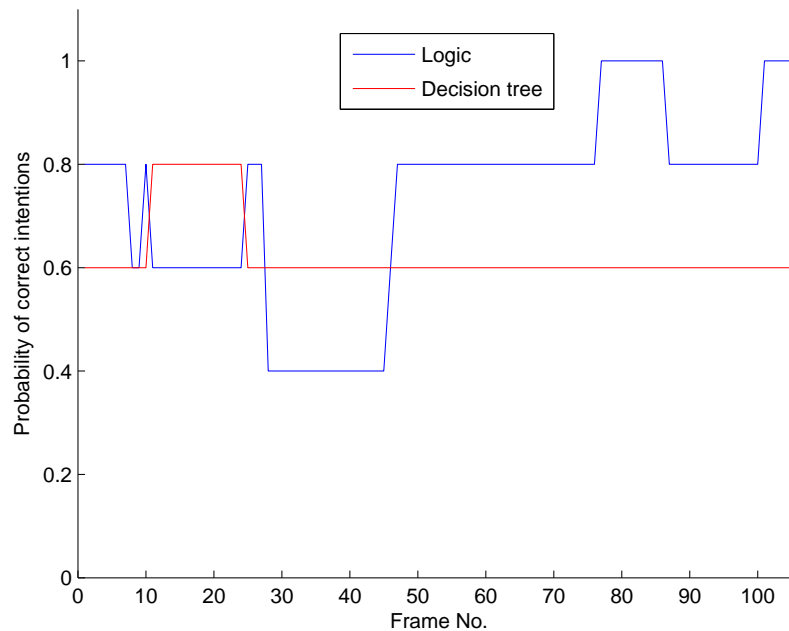


Figure 13: Comparison of *a priori* logic and decision-tree accuracy with sparse data

, $detected_car_track(label_2, t_2, p_2)$ - to the logical car $label_a$. ie there is no direct contradiction between buffer A and B despite the fact that we have identified two differently-labeled tracks with each other.

Essentially buffer B is hierarchically dependent on buffer A in the usual deductive mode, so there are not really any consistency issues. Most of the time, buffer B is simply the logical completion of buffer A. Occasionally there will be a 'damping' if we set buffer B to override A (esp in the offline mode).

Thus, when multiple reported tracks are resolved in buffer B as a single car, there is no direct need to communicate this to buffer A. Usually, the communication to buffer A will presumably be of the following form:

buffer A is a list with slots of approximately the form:

'detected entity' | 'confidence of detection' | 'updated confidence' | 'entity attribute, if any (eg light state)' | time= current

WP4 will thus write an index value to the first column and a real number within [0,1]

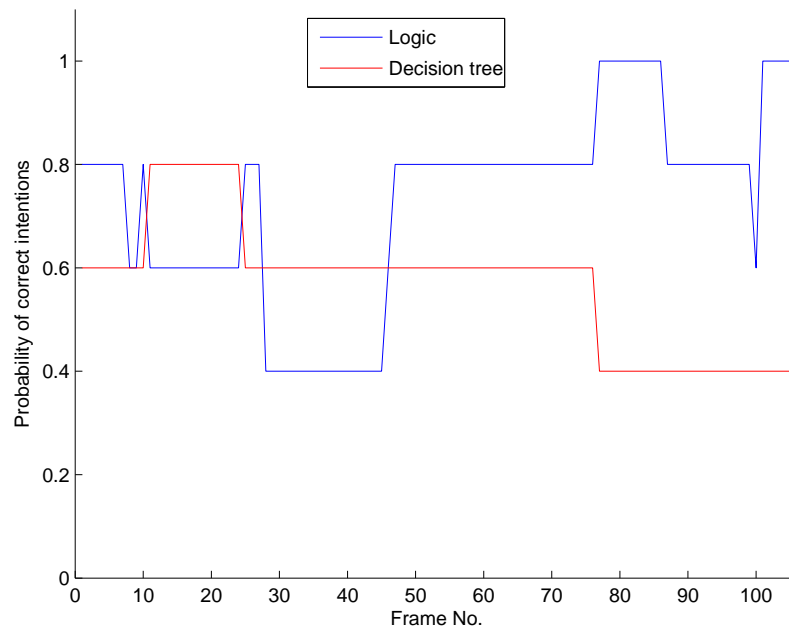


Figure 14: Comparison of *a priori* logic and decision-tree accuracy for non-sparse data

to the second column. WP7SYS will then write (from time to time) a real number within [0,1] to the third column. This represents the updated confidence when all of the context is taken into account. If columns 2 and 3 are persistently different for a long period of time, then this could be an indication that a detector is faulty, or that its confidence (ie the second column) should be tuned down by WP4.

There could also be additional writes of whole entities to buffer A by buffer B for future states relevant to the ECOM intention, in which case it does make sense to have multiple predicate options in buffer A. ie the system may write:

'predicted entity=car in exit lane' | - | 0.3 | - | time = future 'predicted entity=pedestrian in exit lane' | - | 0.2 | - | time = future

(There is obviously much more information about consistent future world states contained in the WP7 system, but only the intention-relevant predicates need to be communicated to buffer A).

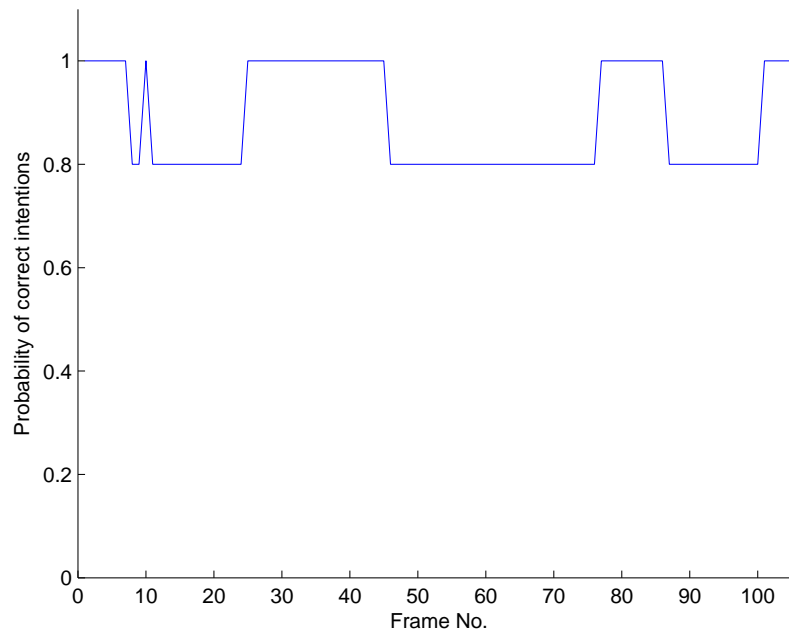


Figure 15: Combination of decision-trees with logical consistency constraints

4.3.2 Buffer structure: temporal considerations

There is no absolute conditional logical dependence between inferred intention and what actually takes place; the DIPLECS car driver can either not perform the actions associated with the inferred intention, or else the world situation can change to modify the intention. In the former case this may be because of a detector error, or else simply a change of plan on the driver's part. In the latter case, perhaps a pedestrian has stepped into the road triggering an 'emergency stop' intention.

In the absence of a priori correlations and labelled data, the only final criterion of the intention "intending to turn left" having existed is the fact that the car is later at the specified place. Because actions (particularly high-level) occur over a long duration, it is thus possible to retrospectively falsify the learned aspects of the 'intentional' model (but not the a priori aspects) in an unsupervised manner. That is, the disparity between inferred states of the ECOM hierarchy and what actually takes place according to the detector inputs potentially allows for some logic-based adaptability via resolution (eg, we can retrospectively identify a sign type based on subsequent action).

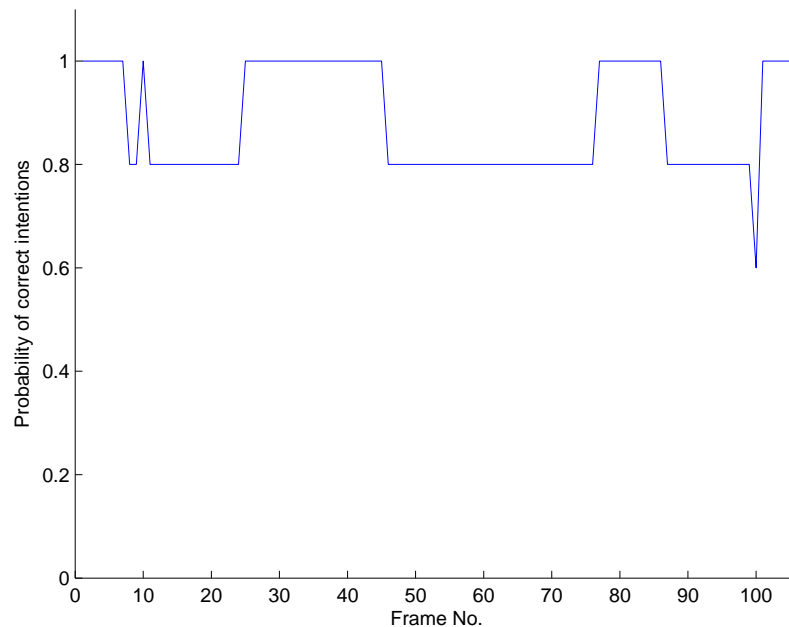


Figure 16: Combination of decision-trees with logical consistency constraints

4.3.3 Buffer structure: multiple consistent world models

In general, we have multiple consistent world models (ie complete predicate sets in Herbrand universes) retained at any one time. 'World-model bifurcation' would be triggered top down by having a critical (ie high-level) detector predicate (such as the junction detector) in an ambiguous state (ie we have 50 percent confidence of being at a junction). Depending on which branch is taken when this is forcibly disambiguated ($prob = 0$ or 1), very different consistent sets are possible.

We are using a mixture of fuzzy/non-fuzzy logic so its possible to force multiple mutually-consistent conditionally-dependent models, eg by asserting in separate buffer A partitions that one or other predicate is true. However, we must, in this case, decide what the triggers for multiple assertions are. The most promising suggestion relates to uncertain high-level predicates eg if we had a 0.5 chance of being at a X-road or T-junction.

These multiple consistent predicate sets would be placed in buffer B; not all of the predicates within the different worlds would need to be communicated to buffer A (see above).

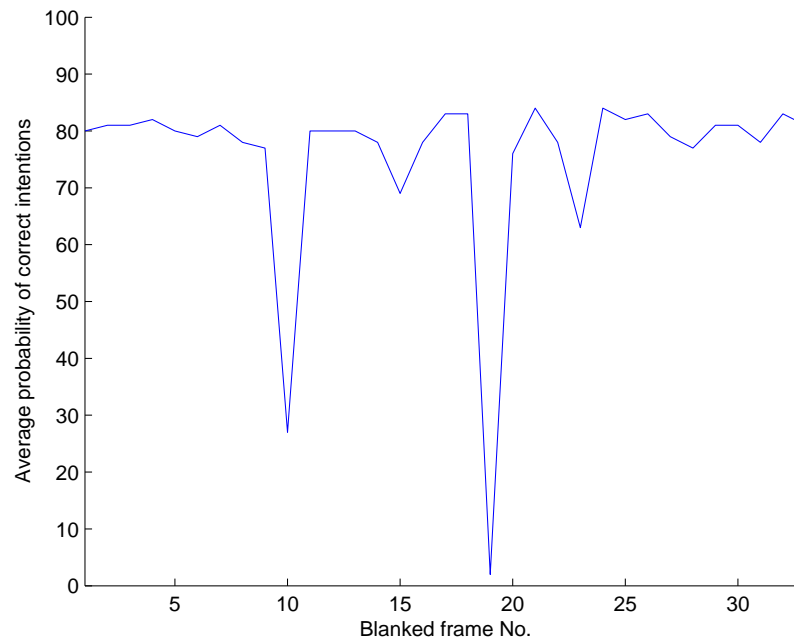


Figure 17: Size of consistency set verses feature number (sparse data)

In general, it is hence important, in interface terms, to distinguish in what sense 'multiple predicate hypotheses' are being used: ie whether it refers to future driver & traffic possibilities, or whether it refers to predicates within different self-consistent worlds (though at the logical level the first notion is implicit within the second, so it only matters in terms of how buffer B interacts with buffer A).

The PODO is thus contained within Buffer B, and the ODO within Buffer A.

5 Work Details: Part 3 - Implementation of a Fuzzy-logic deductive system.

The existing WP7SYS was based on binary logical inputs from vision sensors. The use of binary logic as inputs from a simulated real world sensor implied sensor values with crisp boundaries.

Fuzzy logic, or fuzzy set-theory can be used as a useful tool for the generation of logical consistency amongst predicates even with very sparse and noisy sensor inputs(i.e

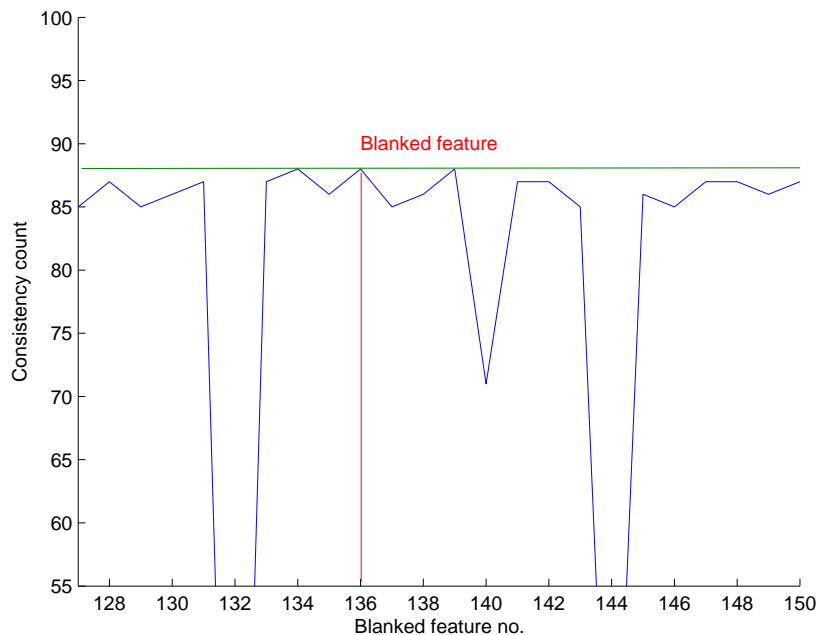


Figure 18: Size of consistency set verses feature number (non sparse data)

vision inputs). It has the advantage of replacing crisp binary values with continuous confidence intervals (at the lowest feature level) [6, 7], thus catering for erroneousness or ambiguity implicit in the sensor values. First-order fuzzy predicates are defined along with their fuzzy membership functions $\psi_z(x) = [0, 1]$ [4]. Hence the process of predicate respecification for top-down bootstrapping of erroneous sensor values using feature blanking technique can be replaced with *down-weighting* confidence intervals of features rather than completely blanking them. Fuzzy logic (i.e. *fuzzy PROLOG*) uses a set of continuous sub-intervals on $[0, 1]$ or a finite union of sub-intervals. A set of discrete *aggregation operators* of type $\Psi : [0, 1]^n \rightarrow [0, 1]$ [4] defined by a generalization that subsumes disjunctive operators (*triangular co-norms; max, sum*) or conjunctive operators (*triangular norms; min, prod*) propagate truth values through the predicate rules and can certainly be used for predicate variable instantiation or recursive rule based resolution of a *PROLOG* query.

Consider the following example; Let X be set of features (sensor inputs), where as x denotes a generic element of X , thus, $X = \{x\}$. Let $\psi_z(x)$ be a membership function that characterizes the fuzzy set Z in X , therefore; $\psi_z(x) = [0, 1] \Rightarrow$ grade of membership of x in Z , if $X = \{x_1, x_2, \dots, x_n\}$ then $P(x_1, x_2, \dots, x_n) \rightarrow \{P(x_1), P(x_2), \dots, P(x_n)\}$ are

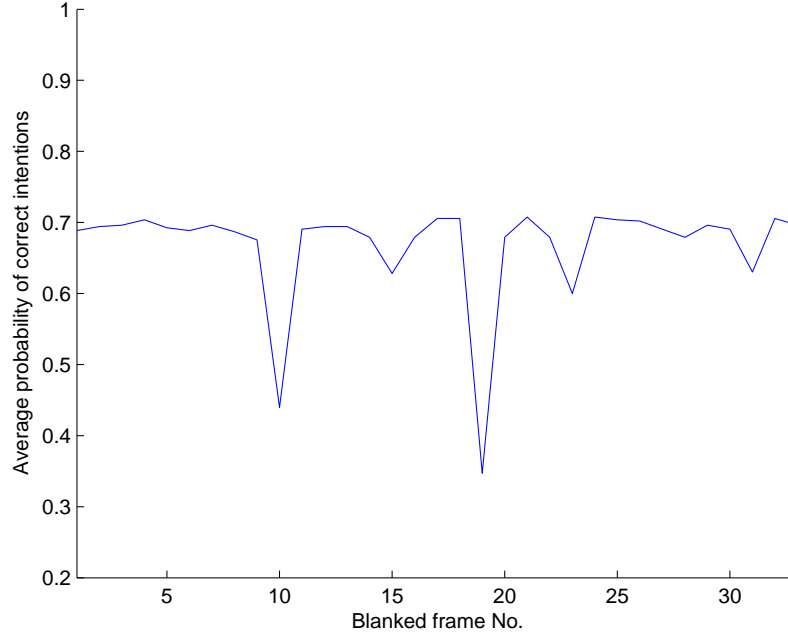


Figure 19: Accuracy verses feature number

the predicates generated from the feature set X , if $\psi_z(x)$ is considered to be a simple ramp function ($\psi_z(x) : Z \rightarrow Z$);

$$\psi_z(x) \equiv \begin{cases} x & \text{for } 1 \geq x \geq 0; \\ 0 & \text{for } 1 < x < 0. \end{cases}$$

A fuzzy predicate $P_{fuzzy}(x_n)$ is defined as,

$$P(x_n) \xrightarrow{\Delta} P_{fuzzy}(x_n) : \{(P(x_n), \psi_z(P(x_n))) | x_n \in (x_1, x_2, \dots, x_n) \wedge \psi_z(x) \in [0, 1]\} \quad (2)$$

where as, Δ is a fuzzy operator.

$$\{detect_road(left, conf_1) \wedge detect_road(opposite, conf_2) \wedge detect_road(driver, conf_3) \wedge junction_type(Type, Conf)\} \quad (3)$$

where as ($conf_1, conf_2, conf_3 \in \psi_z(x)$) are the respective confidence values of each fuzzy predicate, Type & Conf are uninstantiated variables.

$$\{\neg(detect_road(left, conf_1) \wedge detect_road(opposite, conf_2) \wedge detect_road(driver, conf_3)) \vee (junction_type(left_stem_t_junction, Conf) \wedge (Conf : \min(conf_1, conf_2, conf_3)))\} \quad (4)$$

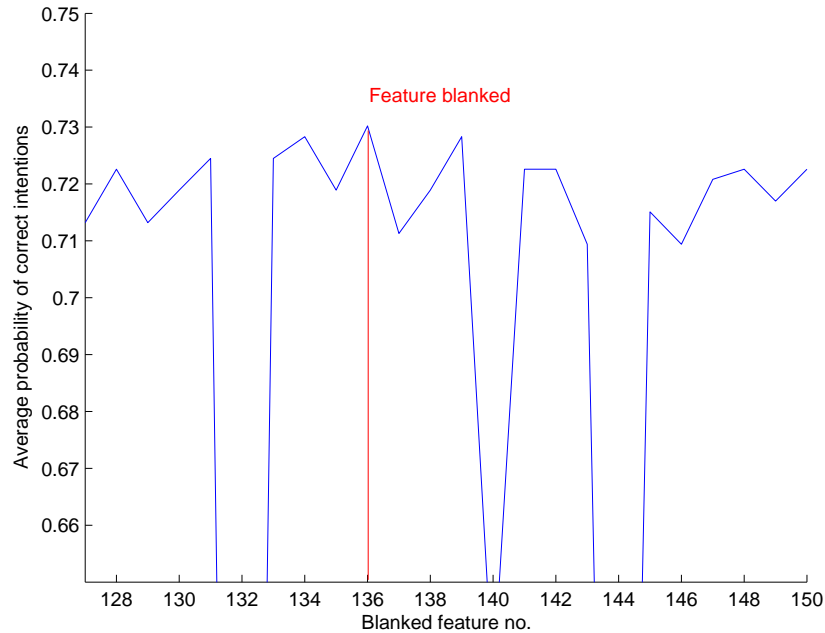


Figure 20: Accuracy verses feature number

'*min*' is a conjunctive operator similar to t-norm. A *PROLOG* query such as: (?- junction_type(Type,Conf)), should lead to a straightforward deductive resolution: (Type \Rightarrow left_stem_t_junction, Conf \Rightarrow [1,0])

The wholesale reimplementation of the existing deductive system within fussy logic that we have thus performed should enable the reconciling of abstract deductions with pre-symbolic features input presented by the various vision and control sensors; in other words, it would address the issues of symbol grounding [2] and symbol tethering [3], and feature respecification (i.e cognitive bootstrapping based on *fuzzy confidence thresholds*).

6 Summary

Three major issues have been addressed to render the previous work (D7.1) applicable in the context of the system as a whole. These are:

1. The Issue of Data Sparsity.

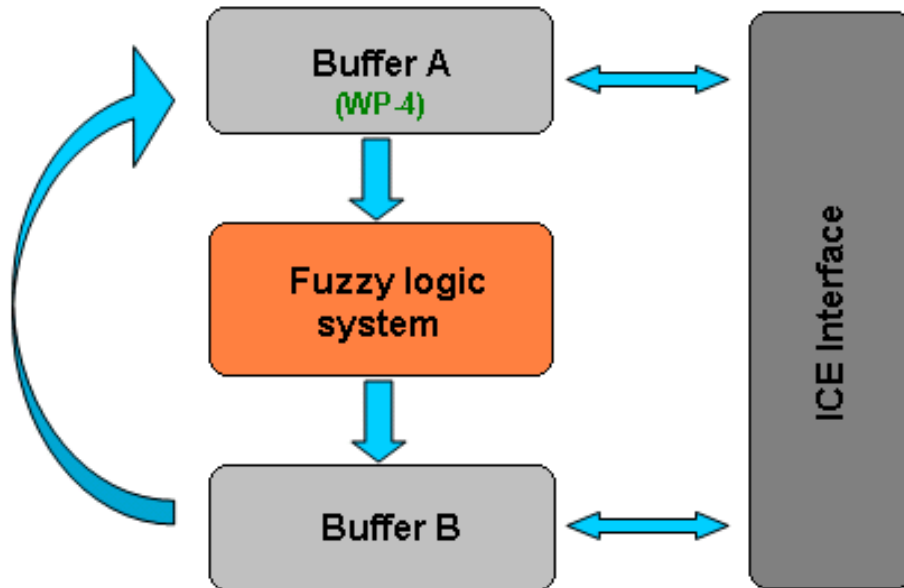


Figure 21: Buffer A - Buffer B relationship

Deliverable D7.1 employed fully ground-truthed input data, defined at all levels of the intentional and scene-description hierarchy. Real input data is likely to be a very sparse subset of this, so the logic system has been tested in the adverse condition of realistic data sparsity.

2. Implementation of a Fuzzy-logic deductive system.

The ground-truthed input data used for Deliverable D7.1 was of a binarised variety. Input data from WP4 has associated confidence values, and the logic system has been extended to accommodate to this.

3. Specification of Interface

The format for interfacing of detector input with the logical deduction system has been fully worked out in term of concrete scenarios, and the logical structure modified accordingly. Issues relating to compatibility with the ICE-interface have also been addressed.

Beyond these objectives, the ongoing refinement and extension of logical predicate/clause structure has continued.

References

- [1] S. Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.
- [2] Stevan Harnad. The symbol grounding problem. *Phys. D*, 42(1-3):335–346, 1990.
- [3] Nick Hawes, Jeremy Wyatt, and Aaron Sloman. An architecture schema for embodied cognitive systems. Technical Report CSR-06-12, University of Birmingham, School of Computer Science, November 2006.
- [4] Susana Muñoz-Hernandez and Wiratna Sari Wiguna. Fuzzy cognitive layer in robocupsoccer. In *IFSA '07: Proceedings of the 12th international Fuzzy Systems Association world congress on Foundations of Fuzzy Logic and Soft Computing*, pages 635–645, Berlin, Heidelberg, 2007. Springer-Verlag.
- [5] A. Sloman. Key ideas of semantic models, implicit definitions and symbol tethering, 2007. Retr. 1/12/7 from: www.cs.bham.ac.uk/research/projects/cogaff/talks/grounding.slides.ps.
- [6] L. A. Zadeh. Fuzzy sets. pages 19–34, 1996.
- [7] L.A. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.