Selecting Ghosts and Queues from a Car Trackers Output using a Spatio-Temporal Query Language

Christian Köhler
Albert-Ludwig-Universität Freiburg
Institut für Informatik
Georges-Köhler-Allee
79110 Freiburg, Germany
ckoehler@informatik.uni-freiburg.de

Abstract

This paper presents a spatio-temporal query language useful for video interpretation and event recognition. The language is suited to describe configurations of objects moving on a plane. To demonstrate its applicability it has been tested on the output of a tracker working on a car traffic scene. The results of two example sets of queries are shown in two videos generated from the trackers data output. The first selects a ghost from the tracking data and the second shows how to find queues of cars in the road traffic scene without prior knowledge of lanes.

1. Introduction

Today a considerably large amount of vision research is focused on detection, recognition and tracking of humans or vehicles. Many successful approaches are purely data driven, while others use stronger models. While detection, recognition and tracking might be solved in a purely data driven manner, image understanding or video interpretation is dependent on more conceptual a priori knowledge built into a vision system. Representing knowledge of a picture or a sequence of pictures by qualitative relations is common in natural language. Relations can be used to describe spatial or temporal knowledge explicitly in a vision system. Qualitative relations are especially useful if no precise quantitative data is available or appropriate (e.g.: content description of a video). This paper introduces a spatio-temporal query language based on qualitative relations. It is explained how the relations are generated from the trackers output data to bridge the gap from quantitative sensor data to symbolic concepts.

1.1. Related Work

To handle the huge amount of data from a sequence of images an internal knowledge representation scheme is needed to reduce the amount of data and finally build a conceptual representation of the processed video stream. In the past two kind of approaches have been proposed to tackle this problem. The first kind of approach uses probabilistic methods to represent knowledge from an image sequence in Hidden Markov Models, Bayesian Belief Networks or Neural Networks (see e.g.: [3], [2]) and might be a natural approach for many people in the computer vision community, since video processing is very sensitive to noisy images. The second kind of approach is based on a declarative a priori representation of knowledge (see e.g: [14], [6]) and might be the more natural approach for people from the artificial intelligence community. In [13] the authors report to use classical filtering techniques to obtain coherent data, that can be associated with symbolic values. Which approach to choose may depend on the concrete vision application and the task to perform. This paper uses a declarative, explicit knowledge representation formalism.

Declarative knowledge representation goes back at least to [11] where networks of constraints are investigated to encode knowledge for vision applications. A lot of progress has been achieved since then, some of it thanks to the tremendous increase of memory capacity and computational speed. Younger work in AI explored knowledge representation schemes for space and time in the field of Qualitative Spatial and Qualitative Temporal Reasoning. The benefit of these knowledge representation schemes is due to the fact, that they encode infinitly many cases into a set of finitly many relevant ones. Some recent work in Qualitative Spatio-Temporal Reasoning examines the complexity of combined spatio-temporal knowledge representation formalisms [8] and investigates the use of qualitative spatio-temporal representations and abduction in an architecture for Cognitive Vision [4].

In the past cognitive systems, represented knowledge explicitly using *chronicles* [6] a temporal representation scheme for time, events and actions. Inspired by this approach, *scenarios* [13] have been used to declare spatio-temporal knowledge in vision applications. Another recent vision system used rules and facts in a *fuzzy metric temporal horn logic* [7].

1.2. Overview

This paper shows how a qualitative spatio-temporal query language is built and how it can be used in a vision application. In section 2 the language is introduced by a choosen set of qualitative relations representing spatio-temporal knowledge of a trackers data output. Section 3 explains the basis algorithm for evaluating queries in the spatio-temporal language, followed by section 4, which illustrates the use by two concrete examples. Section 5 concludes the paper summarizing the gained results. Finally the effect from the two example query sets shown in section 4 can be viewed in two computer generated videos.

2. The Query Language

The vocabulary of the query language consists of unary and binary relations modeling qualitatitive knowledge of distances, orientations, velocities and intervals of time.

The non-temporal relations bind *object variables* in their arguments. Object variables refer to a trajectory generated by the tracker. Each trajectory is given by a position on the ground plane $(x, y)(t)$ (refering to the objects centroid), an orientation $\theta(t)$ and a velocity $v(t)$ for each frame t the object is tracked in the video. Evaluating a relation that binds object variables, leads to a sequence of successive frames from the video, where the qualitative relations holds. These sequences of successive time points are called 'intervals of time' in the ongoing text, neglecting the misuse of the word in a strict mathematical sense. The temporal relations shown in section 2.2 relate intervals of time.

Logical conjunctives are used to express conjuntions and disjunctions of relations as usual. They can be used for both kind of relational expressions: non-temporal and temporal.

2.1. Binary Spatial Relations

In literature spatial knowledge representation schemes are manifold due to the variety of concepts that matter: direction, orientation or topology, just to mention some of the most important. Topology is of special interest for vision applications since it is perspective invariant. In [12] the authors introduce the RCC-8 axioms, which can be used to represent topological knowledge of regions. The calculus can be used to model topological knowledge for regions either in the 2D image or the 3D world. Similarities of qualitative calculi are based on the fact that most of them are substructures of relational algebras [10], [5] in the sense of Tarski [15].

Expressive spatial relations for the tracking data from a video are relations for distance and orientation. Knowledge on orientation always refers to a reference direction. Since the camera perspective in general might change from application to application we only consider orientational relations in the egocentric view from a tracked objects point of view. All distances considered relate two tracked objects by their centroid. Figure 1 gives the idea how spatial relations separate the plane surrounding the tracked object on one view.

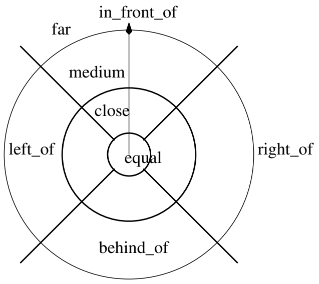

Figure 1: The Model for Spatial Relations

Since the spatial distance relations are not bound to the world coordinate's origin and the orienatational relations are not bound to a global reference direction the vocabulary is reusable for different kind of applications. In other words the proposed set of relations is not bound to a specific camera perspective or specific static objects in the observed scene. On the other hand knowledge can only be expressed by tracked objects related towards each other explicitly.

2.2. Binary Temporal Relations

In [1] the author introduced 13 binary relations to represent knowledge on temporal intervals. The formalism is wide spread and broadly accepted to represent temporal knowledge in various domains.

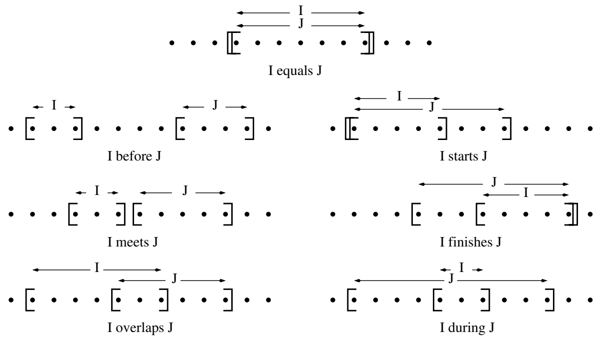

Figure 2: Binary Temporal Relations for Intervals of Time

Figure 2 shows pictoral examples of the binary relations which are used in the query language for two intervals I

and J of time. Converse relations are given by exchanging I and J: `after`, `is_met_by`, `is_started_by`, `is_finished_by`, `is_overlaped_by`, `contains`. The full set of 13 relations is a pairwise mutally disjoint set. Since the modeled temporal domain is discrete (a camera observes the world frame by frame) the operational semantics is different, than in the original Allen Interval Algebra [1], which is defined on a continous temporal domain. More details on the operational semantics for spatial and temporal relations used can be found in [9].

2.3. Unary Relations for Velocity

The difference of moving or static objects oftenly matters to make meaning in natural language. The proposed query language takes this into account by subdividing the measured velocities of a tracked object into three possible classes: `still`, `slow` and `fast`.

2.4. Grounding the symbolic vocabulary

Temporal relations do not need to be grounded since they refer to relational expressions composed by terms of space, velocity and logical conjunctions. These relational expressions are grounded using hard thresholds: Distances are subdivides by 3 thresholds which gives a set of 4 classes: `equal(X,Y)`, `close(X,Y)`, `medium(X,Y)`, `far(X,Y)`, orientations are described by 4 classes refering to a pair of objects in the reference system of the object which is bound in the second argument. The 4 orientational relations are given by: `in_front_of(X,Y)`, `behind_of(X,Y)`, `right_of(X,Y)` and `left_of(X,Y)`. Unary relations for velocity are given by: `still(X)`, `slow(X)` and `fast(X)`.

3. Evaluation of a query

In the current version of the implemented system each query is given by a tree. Each node in the tree represents a relation. Childs of a tree node are the relations arguments. The binary spatial relations and the unary relations for velocities are bound to object variables as arguments. Binary temporal relations take terms build from spatial relations, relations on velocity and logical conjunctives as arguments. Temporal binary relations can be combined using logical conjunctives as well. Evaluating the query starts with generating all possible object variable bindings which is combinatory in the number of tracked objects. For each possible binding the tree is evaluated starting from the leafs of the tree (spatial and velocity expressions) propagating the resulting intervals of time to the parent nodes. If finally the root node is evaluated and the resulting set of temporal intervals is not empty the query holds for each frame covered by the intervals on the bound objects.

4. Experiments and Results

The original video sequence consist of 2060 frames grabbed at a rate of 20 frames per second. 25 objects are tracked. Each object trajectory is given by a tuple $(x, y, \theta, v)(t)$ for each frame t where the object is tracked.

4.1. System Overview

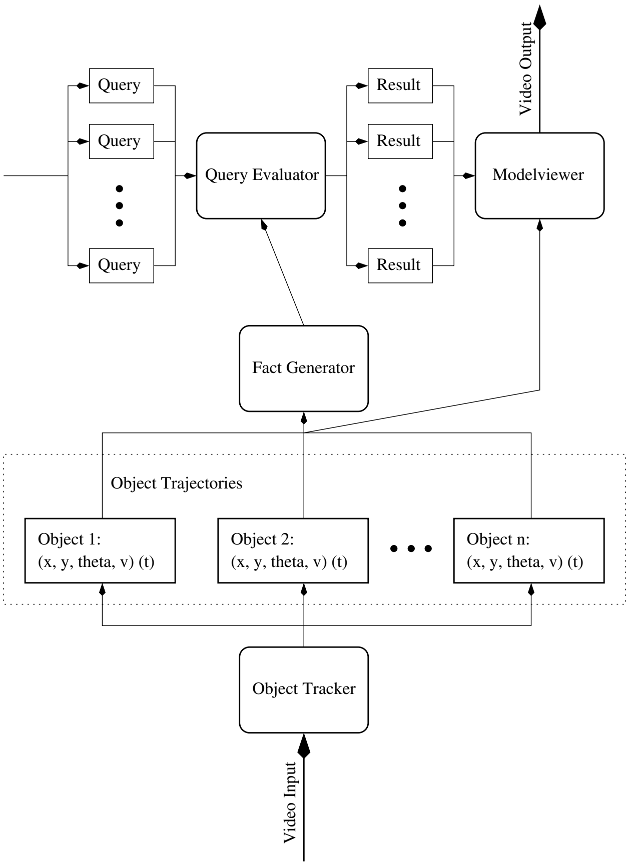

Figure 3: Sytem Architecture (Main Components)

Figure 3 shows the main components of the used system. Starting from the video intput stream the *Object Tracker* extracts trajectories of cars from the observed scene. The *Fact Generator* generates the grounded symbolic relations from the trajectory data for each pair of objects in the case of the binary and each object of the unary relations. The grounding of relations is currently done in a preprocessing step. Queries are evaluated in the *Query Evaluator* bottom up as explained in 3. Ground instances of non-temporal relations used in the Query Evaluator are produced by the *Fact Generator*. The results from query evaluation are sent to the *Modelviewer*. The *Modelviewer* renders the scene from the car trajectories in a 3D model. Each car tracked is shown as a sphere with a diameter of 5m in the 3D model. The default color of each sphere is a medium blue. To illustrate the results of queries, all objects X selected by a query expression are colored in a unique marker color different from the default color for each frame, where the query expression holds. If two or more query expressions hold at the same frame for the same object X, the color of the first matching query is assigned. The resulting constructed 3D scene over all frames is written to the video output to show the effect of the spatio-temporal query languages expressions over time.

4.2. Parameters for the Qualitative Relations

As explained in 2.4 the qualitative relations built by the fact generator in the preprocessing step are grounded on hard thresholds. Figure 4 shows the used values for the thresholds. A car at a speed of less than $1m/s$ is considered as still standing since still standing objects move due to sensor noise. The front opening angle is chosen to be small to associate pairs of objects on one lane to a single queue. The angle for *behind_of* a car is broader to select ghost cars with the formula from 6, which might change their orientation after tracking is lost.

Distance	
equal	$[0m, 1m[$
close	$[1m, 5m[$
medium	$[5m, 15m[$
far	$[15m, \infty m[$

Velocity	
still	$[0m/s, 1m/s[$
slow	$[1m/s, 3m/s[$
fast	$[3m/s, \infty m/s[$

Orientation	
in_front_of	$]-18°, +18°[$
behind_of	$]+135°, +180°] \cup]-180°, -135°[$
right_of	$[-135°, -18°]$
left_of	$[+18°, +135°]$

Figure 4: Parameters used for Symbol Grounding

4.3. Selecting Ghosts from the Traffic Scene

No matter how good a tracker works, it will hardly be 100% accurate. When tracking is lost but enough evidence is found a tracker might get stuck on a hypothesis for an object that already left the place. In the first result video this happens due to partial occlusion: the big car number 2 occludes car number 1, which can be seen in the left side of figure 7, afterwards when the queue of cars starts to move tracking is lost.

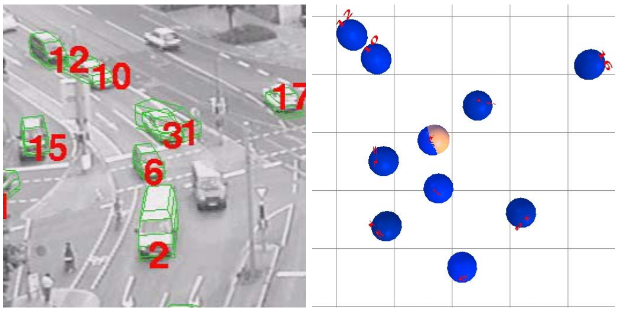

Figure 5: Tracker and Query Results in Frame 1548

When a car is lost from tracking it is not possible to detect this based on the trajectory data output of the tracker, unless another car runs through the lost object from behind to the front as shown in 5. A sequence of successive frames where a still standing car X is run over by another car A can be expressed by a conjunction in terms of the spatio-temporal query language as shown in the first formula in figure 6. The second more general case describes a sequence, where car A runs into a still standing car X from behind. Since the equal distance threshold is only 1m we might conclude in both cases, that the still standing object X is a ghost based on the fact that cars are rigid objects. The video generated from this example query set shows the results of both queries: all cars X matching the first expression are marked red, for the interval of time where the expression holds. Cars X which match the second expression are marked orange in the generated video. Figure 5 is this a representative snapshot grabbed from this video. Since the first expression is always marked first by the modelviewer and the second is more general than the first, a car X will always be marked red for intervals of time, where both formula hold. The most general case, which makes a ghost selectable based on the assumption that no pair of objects can be at the same place at the same time and ghosts do not move can be described by $\{\mathtt{still}(X), \mathtt{equal}(A, X)\}$ in terms of the query language.

```
I: { still(X), behind_of(A,X), close(A,X) }
J: { still(X), equal(A,X) }
K: { still(X), in_front_of(A,X), close(A,X) }
I meets J, J meets K

I: { still(X), behind_of(A,X), close(A,X) }
J: { still(X), equal(A,X) }
I meets J
```

Figure 6: Set of Queries for selecting Ghosts from the Tracking Data

4.4. Selecting Queues from the Traffic Scene

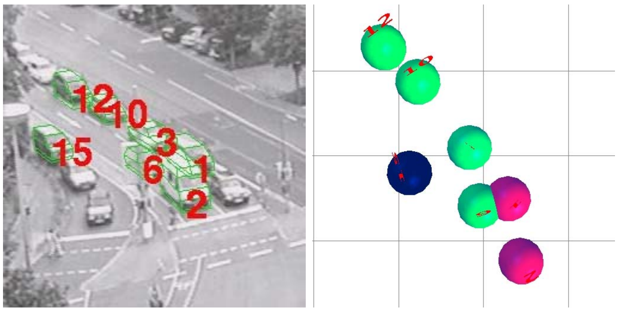

Figure 7: Tracker and Query Results in Frame 1260

With the relations defined on the trajectory data, it is possible to build queues from head and tail describing the alignment of objects over intervals of time. An example for this is shown in figure 7. The set of formula shown in figure 8 was used to generate the second example video. The image on the right of figure 7 was grabbed from this video. First queue tails are marked in green, then queue heads are marked in red. All objects not marked as head or tail of a queue are colored based on the thresholds for velocity: dark blue for still standing, a medium blue for slow and a light blue for fast moving objects.

```
{ still(X), still(A), in_front_of(X,A),
  behind_of(A,X), close(A,X) | medium(A,X) }
{ still(X), still(A), in_front_of(A,X),
  behind_of(X,A), close(A,X) | medium(A,X) }
{ still(X) }
{ slow(X) }
{ fast(X) }
```

Figure 8: Set of Queries for selecting Queues from the Tracking Data

To see the effect of the query set over time its strongly recommended to watch the generated videos.

4.5. Runtime and Memory Consumption

The selected unary and binary relations fit well to give meaning in the expamples. In other domains other sets of relations might be of interest. Therefore the focus of the implementation was on reusability and extendability of the query evaluation mechanism and the query language itself. The spatio-temporal query language was implemented in Java. While evaluating a query: no symmetry of relations is exploited, no constraint propagation method is used for speed up, no sophisticated methods from computational geometry were applied to calculate the symbol grounding for the binary spatial relations and no strategy on the ordering of literals of the constraint expressions was used. The implementation used a straight forward bottom up query evaluation on all possible bindings for the object variables.

The reference machine used a Athlon 1.6 GHz CPU. The preprocessing, runs in $O(n^2)$, where n is the number of tracked objects. It takes 2400ms–2500ms to compute on the reference machine. (An exact runtime is hard to give due to memory manegement in the java virtual machine, which causes varying runtimes executing the same code on the same data. All observed runtimes, felt into this upper and lower bound). Memory consumption of the preprocessing step can be described by the number of produced unary and binary relations holding for an interval of time. 1777 unary and binary relations where produced, each of them stores 2 integers (start and end frame where the relation holds) of memory. The memory consumption is quadratic in the number of tracked objects due to the fact that the relations generated are binary relations over a finite set of objects.

Thanks to preprocessing that grounds the symbolic vocabulary on all pairs of object variables, the evaluation of a query takes only a little amount of time. Each query from the two given example sets, takes less than 120ms to compute. This result is not astonishing, since the queries from the two examples sets are built from conjunctions containing not more than 2 unbound object variables.

Summing up runtime and memory consumption results it can be said, that using the given set of queries on a video of 100 seconds with a framerate of 20 frames per second is already possible in realtime on a current standard PC without sophisticated implementation techniques for query evaluation on a limited set of objects tracked (in the shown examples 25 during the 100 seconds). Increasing the number of object variables in a query will cause a combinatory explosion of the runtime.

5. Summary and Conclusions

The number of objects tracked in a frame will always be limited, since the picture itself is limited in its size. This leads to two interesting open questions concluding the remarks on runtime and memory consumptions:

- What relational expressions do actually need more than 2 objects, that are related without being able to express an equivalent expression with a conjunction of 2 expressions?
- Which applications do actually need to store more than a small set of tracked objects over a period of time to build the intended meaning of a video input stream?

Positive answers to both questions will lead to an increase of runtime for query evaluation. 3 or 4 different object variables in a query might still be reasonable for a limited set of objects.

Based on given a priori knowledge changing configurations of objects over time can be expressed by the spatio-temporal query language. The language can be used to build a conceptual representation of the video or generate events from spatio-temporal configurations of objects tracked in a video stream. Only very small spatio-temporal formula were necessary in the given examples. All parameters are plausible based on common-sense knowledge in physics and geometry, they are easy to guess and do not have to be too precise to make good results. The vocabulary chosen is simple and close to natural language, which helps to design a set of queries for a given task. False positives from tracking can be identified by constellations, that are physically impossible. Using the proposed spatio-temporal query language to assist an object tracker might improve tracking, since small expressions are fast to evaluate and represent common-sense knowledge of geometry and physics in qualitative terms. Due to the fact that no geometry of the surrounding environment was taken into account, the language is reusable by other applications.

References

[1] J. F. Allen: "Maintaining Knowledge about Temporal Intervals", *Communications of the ACM*, Vol. 26, No. 11, pp. 832–843, November 1983.

[2] A. J. Howell and H. Buxon: "Active Vision Techniques for Visually Mediated Interaction", *Presentation: ICPR '02, International Conference on Pattern Recognition*, Quebec City, Canada, August 2002.

[3] H. Buxton and S. Gong: "Advanced Visual Surveillance using Bayesian Networks", *International Conference on Computer Vision*, Cambridge, Massachusetts, June 1995.

[4] A. Cohn, D. R. Magee, A Galata, D. C. Hogg and S. M. Hazarika: "Towards an Architecture for Cognitive Vision using Qualitative Spatio-Temporal Representations and Abduction", In: Ch. Freksa, C. Habel, K. F. Wender (Eds.) *Spatial Cognition III* Tutzing Castle, Lake Starnberg, May 2002.

[5] I. Düntsch: "A tutorial on relation algebras and their application in spatial reasoning", Given at *Cosit '99*, `http://www.cosc.brocku.ca/~duentsch/archive/relspat.pdf`, 1999.

[6] M. Ghallab: "On Chronicles: Representation, On-line Recognition and Learning", Aiello, Doyle and Shapiro (Eds.), *Principles of Knowledge Representation and Reasoning*, p. 597–606, Morgan-Kauffman, November 1996.

[7] R. Gerber, H.-H. Nagel and H. Schreiber: "Deriving Textual Descriptions of Road Traffic Queues from Video Sequences", In: F. van Harmelen (Ed.) *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI-2002)*, pp. 736–740, IOS Press, Amsterdam 2002.

[8] A. Gerevini and B. Nebel: "Qualitative Spatio-Temporal Reasoning with RCC-8 and Allen's Interval Calculus: Computational Complexity", *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI'02)*, 2002.

[9] C. Köhler, "A Proposal for a Spatio-Temporal Qualitative Query Language based on Primitive Geometry for Cognitive Vision Systems", Albert–Ludwigs–University Freiburg, Chair for Foundations of Artificial Intelligence, Georges–Köhler–Allee Geb. 52, D–79110 Freiburg im Breisgau, Germany, `http://www.informatik.uni-freiburg.de/~cogvisys`. 2003.

[10] P. B. Ladkin and R. D. Maddux, "On Binary Constraint Problems", *Journal of the ACM*, Vol. 41, No. 3, pp. 435–469, May 1994.

[11] U. Montanari, "Network of Constraints", *Information Sciences*, Vol.: 7, pp. 95–132, 1974.

[12] D. A. Randell and Z. Cui and A. Cohn: "A Spatial Logic Based on Regions and Connection", In: Bernhard Nebel and Charles Rich and William Swartout (Eds.): *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92).* , Morgan Kaufmann, San Mateo, California, pp. 165–176, 1992.

[13] V. Vu, F. Brémond and M. Thonnat, "'Automatic Video Interpretation: A Novel Algorithm for Temporal Scenario Recognition.", *The Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'03)*, Acapulco, Mexico, 9-15 August 2003.

[14] N. Rota and M. Thonnat: "Activity Recognition from Video Sequences using Declarative Models", *14th European Conference on Artificial Intelligence (ECAI 2000)*, Berlin, Germany, August 2000

[15] Tarski, A.: "On the calculus of relations", *Journal of Symbolic Logic*, Vol. 6, pp. 73–89 1941.