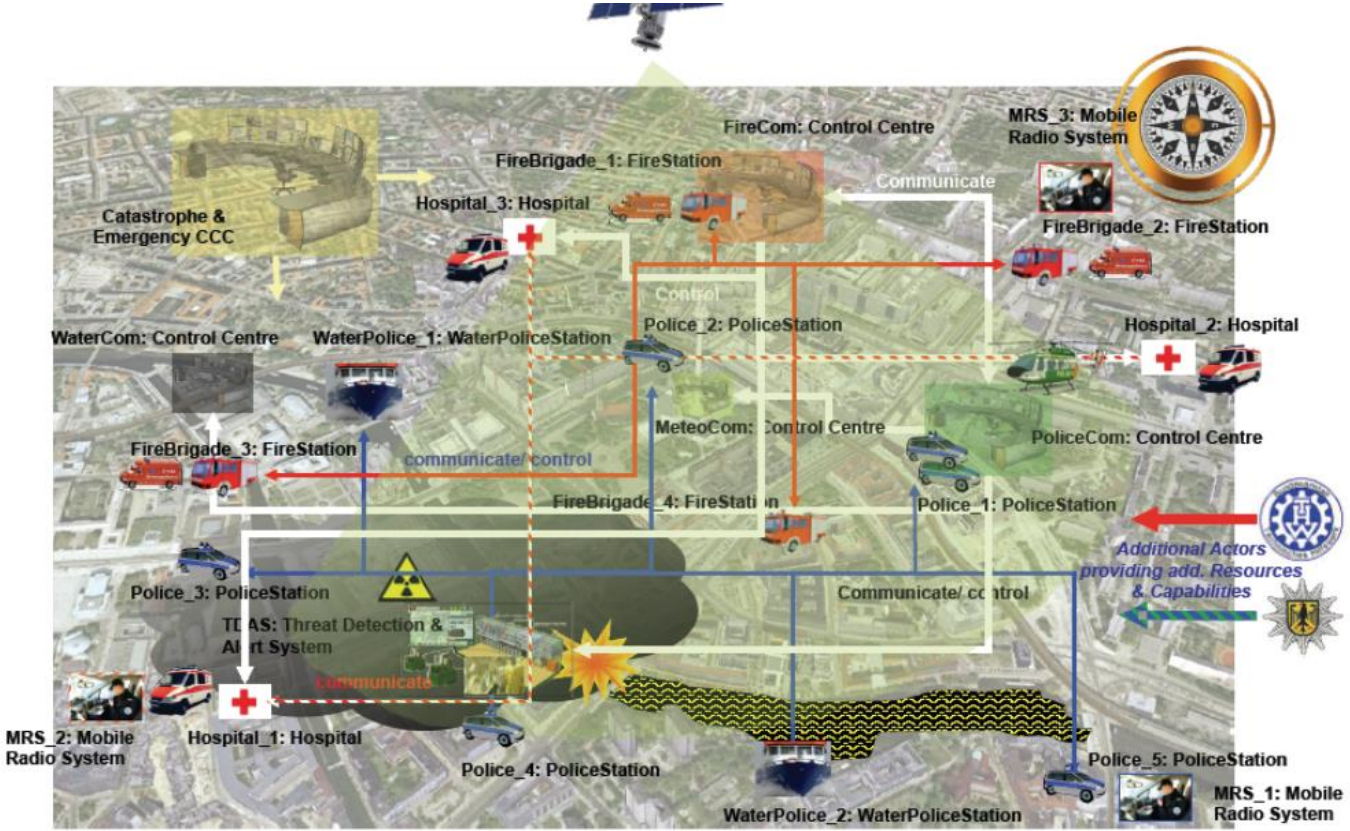
# Beliefs, Confidence, and Ground Truth: What may go wrong in distributed cooperative decision making?


## Werner Damm

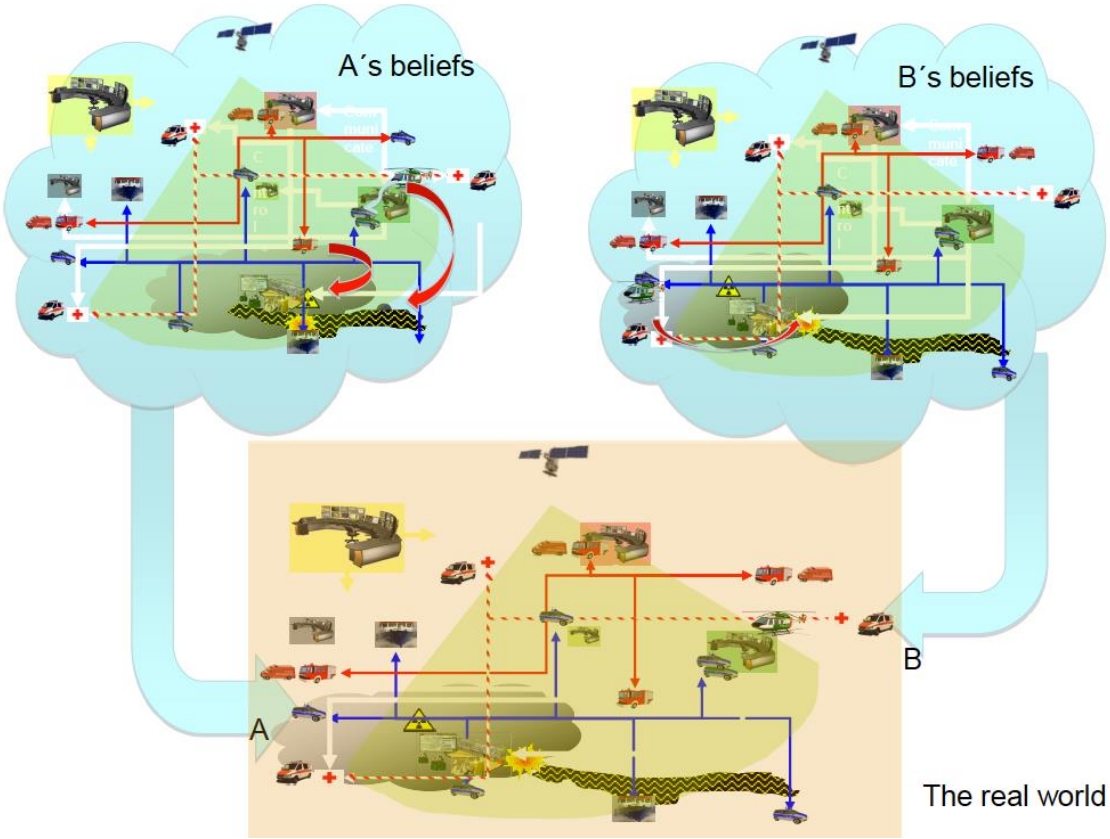
Director, Interdisciplenary Research Center on Critical Systems Engineering, Carl von Ossietzky University Oldenburg
Chairman, OFFIS Transportation
Chairman, SafeTRANS
Member acatech

Joint work with
Alberto Sangiovanni-Vincentelli, UCB
Willem Hagemann, Paul Kröger, Carl von Ossietzky University Oldenburg

# A running Example: Ground Truth in London



# Ground Truth and Beliefs

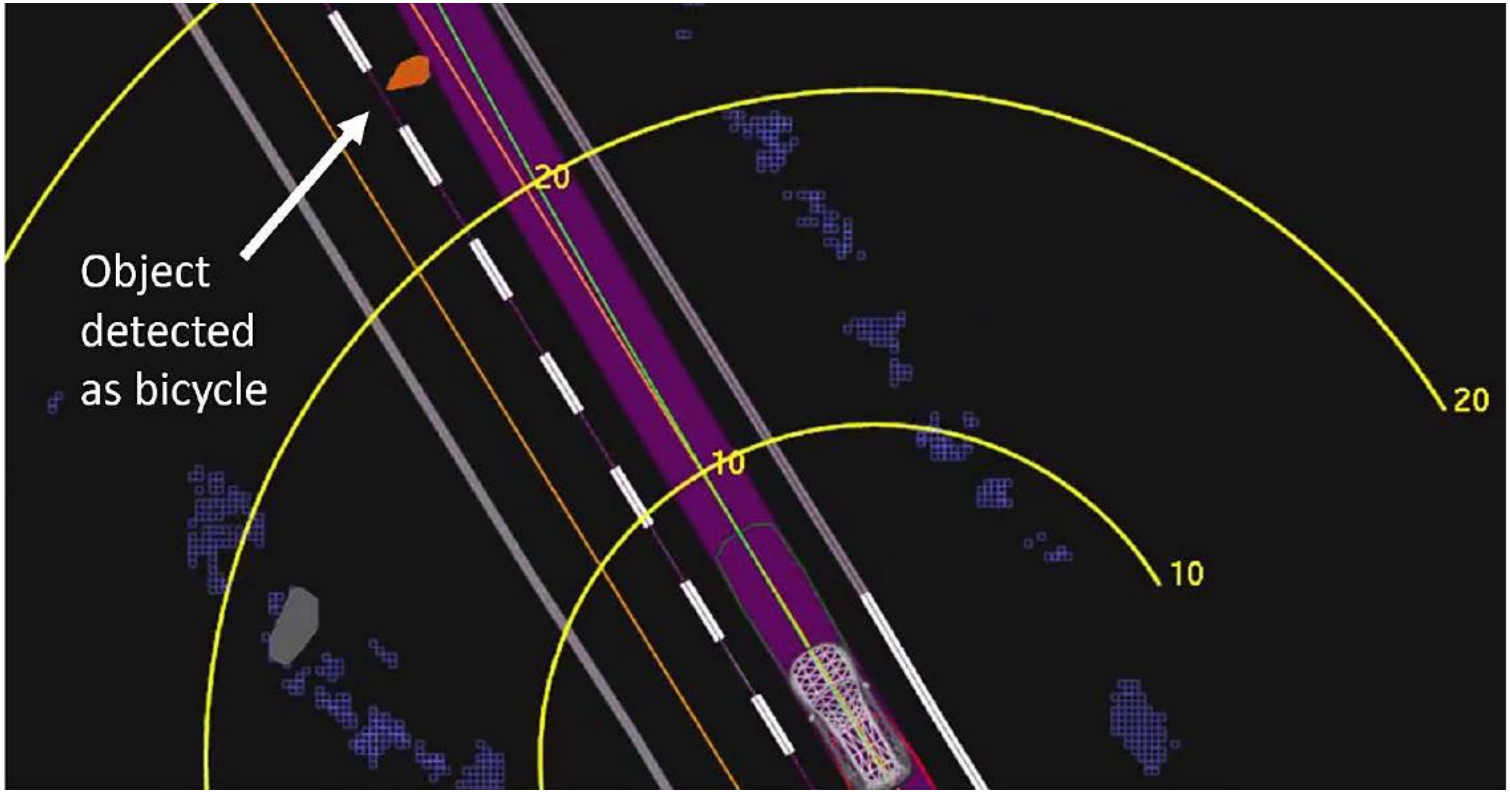


# The discrepancy between what the system beliefs to be true and ground truth can be a matter of life or death

14.09.1993 -

Aircraft thought it was still airborne, because only two tons weight lasted on the wheels due to a strong side wind and the landing maneuver. The computer did not allow braking.  The plane ran over the runway into a rampart.

# What the UBER car believed to be true I



# What the UBER car believed to be true II

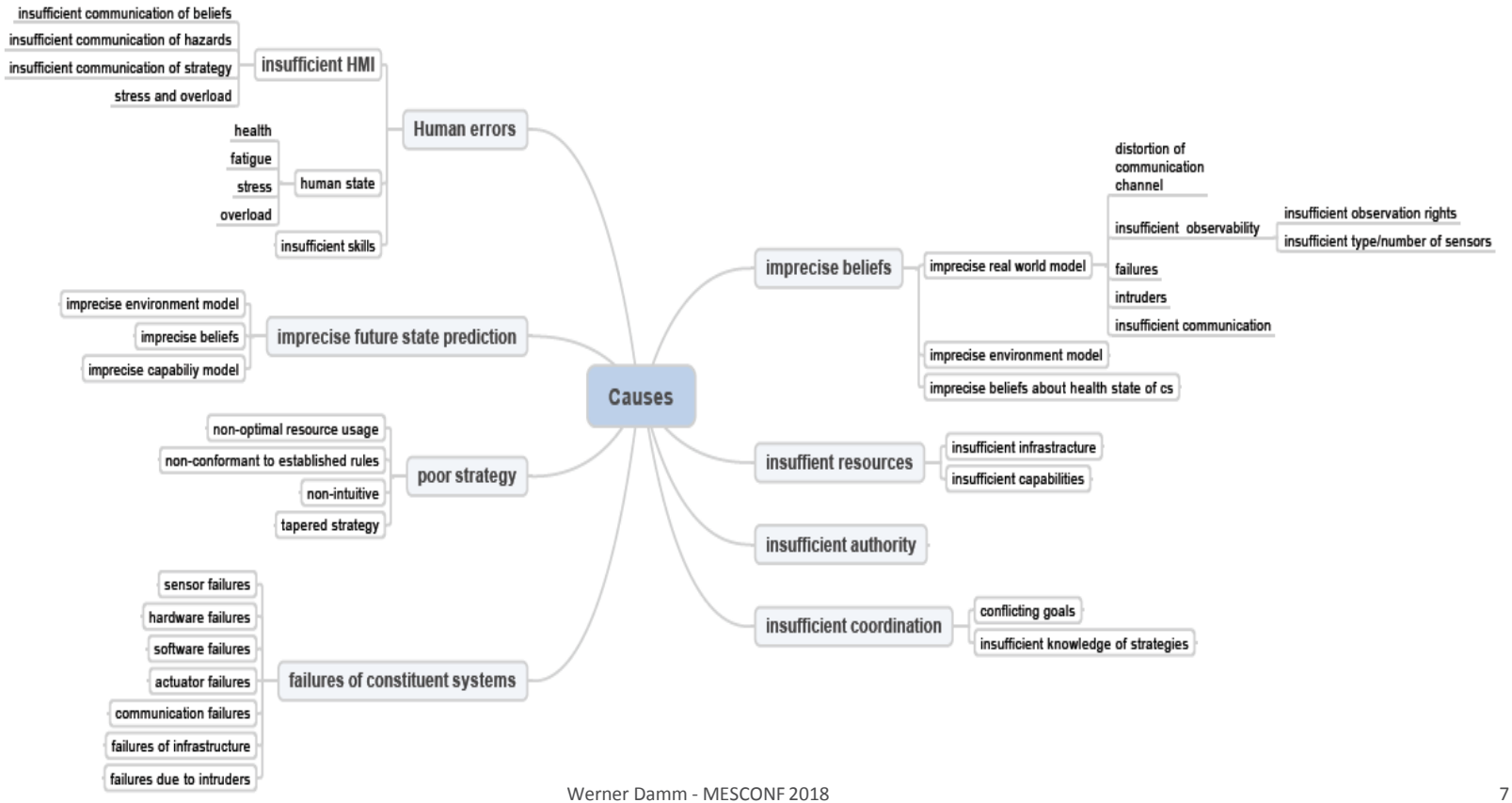
„According to data obtained from the self-driving system, the system first registered radar and LIDAR observations of the pedestrian about 6 seconds before impact, when the vehicle was traveling at 43 mph.

As the vehicle and pedestrian paths converged, the self-driving system software classified the pedestrian

1. as an unknown object,
2. as a vehicle,
3. and then as a bicycle

with varying expectations of future travel path.“

# What may go wrong in distributed cooperative decision making?



# Focus of this talk

1. Perception, Beliefs, and Ground Truth: A specification of the quality of perception
2. Goals, capabilities, strategies: a game theoretic model of Systems of Systems operating under imperfect information
3. Conclusion

# Challenges in the perception chain

How can we assure, that an actor´s belief about its environment is “sufficiently precise” for achieving its services?

- can we observe all relevant artefeacts of the environment?
- can we provide confidence guarantees for artefact identification along the sensor chain?
- even in the presence of failures of relevant subsystems?

# Wanted

**a “provable” robust abstraction relation between the relevant real-world artifacts and the internal digital world model of each system**:

whenever real-world artefact a is “relevant”:

p(a) is true in real world at time t

iff

with high probability $p_{\varepsilon}(a)$ is true in believed world model at time t± **Δ**

# Spoiling Factors

inherent limitations of different types of sensors

> typically compensated by sensor fusion

inherent limitations of object identification algorithms

> either good in recognizing a if a is in real world

> or good in recognizing that a is not present in real world

> Possibly contradicting classifications of objects

# Resulting research topics

- characterize “relevant”: how do we determine those artefacts of the real world (see Damm&Finkbeiner 2015)
  - which must be observed
  - whose existence and evolution is irrelevant for the system´s goals
- precise bounds on epsilon-delta along the complete chain from raw sensor data through sensor fusion through object recognition
- work with two world models (Dempster&Schaefer)
  - safe approximation of existence
  - safe approximation of non-existence
- Let humans help to resolve uncertainty (see Automate project)
- Let control strategies be adapted to uncertainty (see Damm&Fränzle 2018)

# Resulting Research Questions

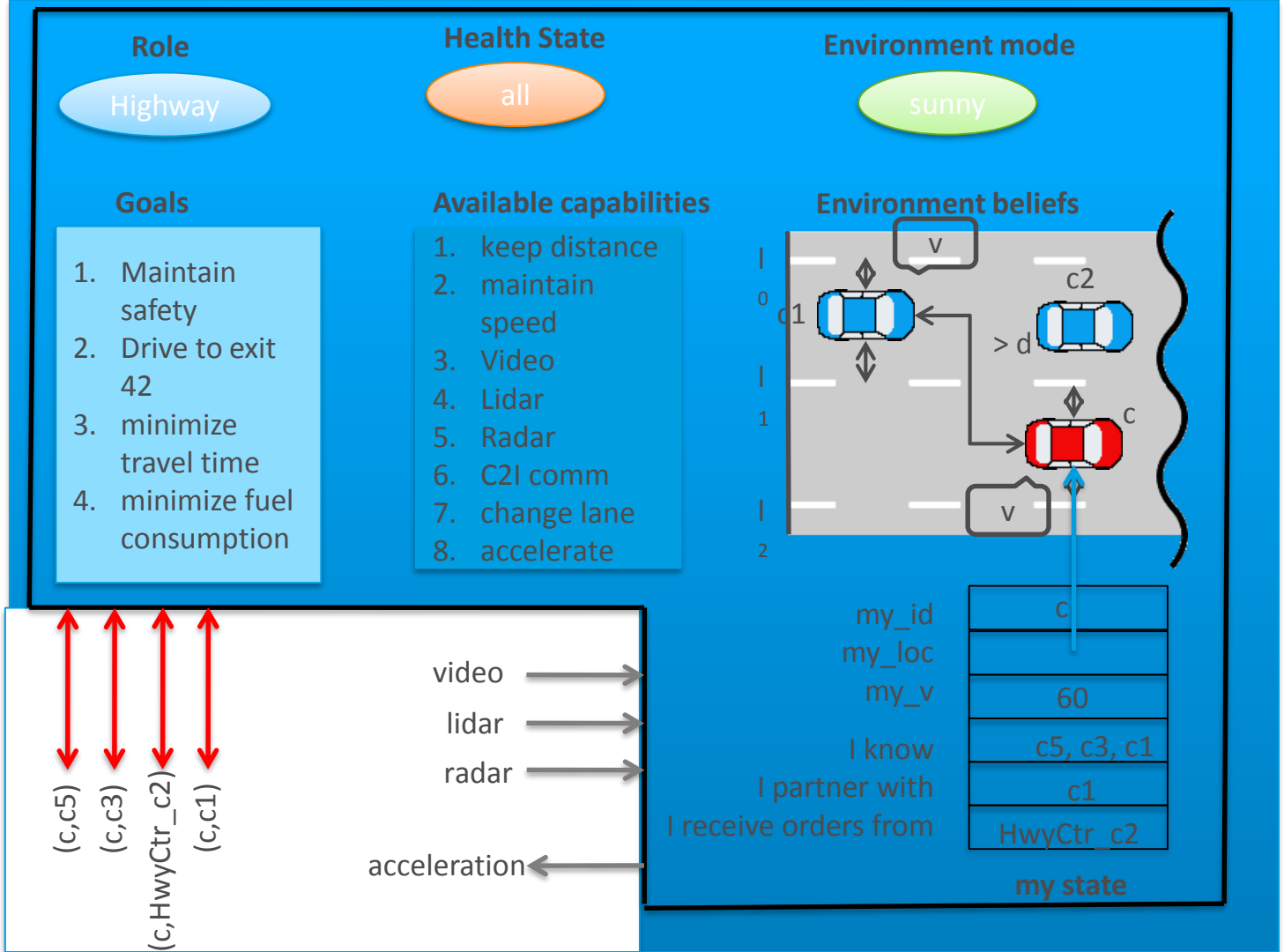
Can we provide probabilistic guarantees for learning algorithms in allowed real-world contexts?

Can we extend heuristic methods such as Hazop analysis to guide search for possibly relevant real-world artifacts

> see code of practice from Prevent Project and

> approach to AI based and formal methods based learning of hazards for highly automated driving in forthcoming V&V project, Damm&Galbas 2018

# TOWARDS A GAME THEORETIC FORMALIZATION


Role
Highway
Health State
all
Environment mode
sunny
Goals
1. Maintain safety
2. Drive to exit 42
3. minimize travel time
4. minimize fuel consumption
Available capabilities
1. keep distance
2. maintain speed
3. Video
4. Lidar
5. Radar
6. C2I comm
7. change lane
8. accelerate
Environment beliefs
v
c2
c1
> d
c
v
l 0
l 1
l 2
(c,c5)
(c,c3)
(c,HwyCtr_c2)
(c,c1)
video
lidar
radar
acceleration
my_id
my_loc
my_v
I know
I partner with
I receive orders from
c
60
c5, c3, c1
c1
HwyCtr_c2
my state

# Observables

For each instance *S:Cl (with Cl∈CL)* of a constituent system *S* of class *Cl* there is a predefined variable *my_id(S):Id* which gives the unique identity of *S*

Systems are part of a physical world.

Let *Ext(Cl)* denote the finite set of extensions of instances of class *Cl* in the real world. For each signal *v Ext(Cl)* and each instance *S:Cl* with identity *i* we denote by *i.v* the *v extension of this instance in the physical environment*.

Examples: position, speeed, weight, temperature, …

Beliefs …

# Beliefs

For each constituent system *S* of **S**, let $V_{ENV}(S) \subseteq V_{ENV}(\mathbf{S})$ be the subset of environment variables *currently observed by S*;

this includes *my_pos* and *my_vel* for all extensions *v* of instances of class *Cl* relative to the global coordinate system of **S**.

The valuation of these signals will only be perceived through sensors, sensor fusion, and merging of these views with other constituent views of their environment.

Hence, we refer to a valuation as measured or obtained through sensor fusion or belief fusion as *beliefs of S about its environment.*

# Beliefs need not be true …

Beliefs about the real world, and the real world itself may in principle differ arbitrarily.

We distinguish between the valuation of variables of observation predicates

- as seen by an omniscient observer $obs_{RW}(v)(t)$
- as seen by the system S $obs(S)_{ENV}(v)(t)$

Ideally these are “sufficiently similar” for all “relevant” environment variables up to bounded errors.

# Beliefs

For example, a car may have a distorted view of its position due to a temporal distortion of its GPS system, and thus its belief about its position (the current value of its local signal *my_pos*) can differ from *i.pos* (where *i* is the identity of this car).

Such beliefs about the Environment are represented through local variables of a system. These include variables my_pos, my_v, … for all physical extensions of S, and of such variables of systems S´ “relavant” to S.

For $v \in V_{ENV}(S)$ we denote by $obs(S)_{ENV}(v)(t)$ the belief *S* has about the value of signal *v* at time *t*.

# Beliefs (cont.)

*we denote by*

$$Beliefs(S)(t)$$

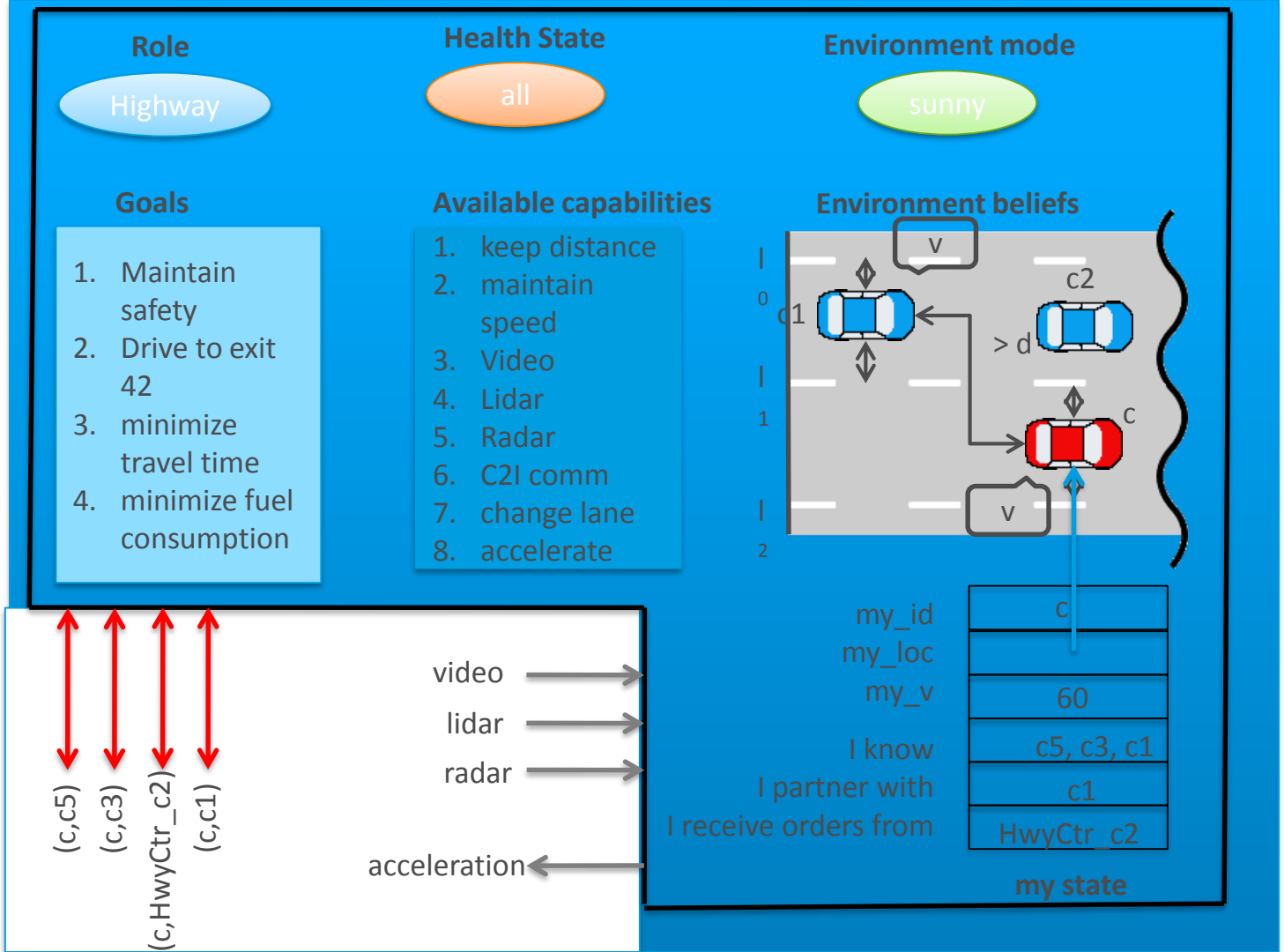
*the subset of predicates of P(S) which S believes to be true at time t, i.e.*

$$Beliefs(S)(t) = \{p \in P(S) \mid [[p]]obs(S)_{ENV}(v)(t) = true\}.$$

*where*

$$[[p]]\ obs(S)_{ENV}(t)$$

*denotes the truth value of predicate p when evaluating its free variables in its local state given by state* $obs(S)_{ENV}(t)$


Role
Highway
Health State
all
Environment mode
sunny
Goals
1. Maintain safety
2. Drive to exit 42
3. minimize travel time
4. minimize fuel consumption
Available capabilities
1. keep distance
2. maintain speed
3. Video
4. Lidar
5. Radar
6. C2I comm
7. change lane
8. accelerate
Environment beliefs
v
c2
c1
> d
c
v
l0
l1
l2
(c,c5)
(c,c3)
(c,HwyCtr_c2)
(c,c1)
video
lidar
radar
acceleration
my_id
my_loc
my_v
I know
I partner with
I receive orders from
c
60
c5, c3, c1
c1
HwyCtr_c2
my state

# Beliefs (examples)

*“I believe that there is a car ahead of me with distance d travelling with speed v”*

*i:Car (d(i.pos,my_pos)=d  i.lane=my_lane  i.speed=v)*

*I believe that there is an object ahead of me with distance d travelling with speed v*

*Cl:CL  i:Cl (d(i.pos,my_pos)=d  i.lane=my_lane  i.speed=v))*

# Robustness

*Robustness* is defined with respect to a function

$$\mathcal{E}(S): V_{ENV}(S) \rightarrow R+ \rightarrow R$$

that specifies the tolerance with which *S* is expecting to be able to measure the real world value of an environment variable at time *t*.

# Robust Truth of Beliefs

We distinguish between

- beliefs of a system about values of its own physical extension as well as of "relevant" neighboring systems as given by $obs(S)_{ENV}(v)(t)$
- the values of these as seen by an omniscient observer $obs_{RW}(v)(t)$

A belief *p of S is true at time t in the real world with robustness* $\mathcal{E}(S)$ *iff*

$$\forall\, v \;\; free(p) \;\; obs_{ENV}(S)(v)(t) \in [obs_{RW}(v)(t) - \mathcal{E}(S)(v,t), obs_{RW}(v)(t) + \mathcal{E}(S)(v,t)\,]$$

$$\Rightarrow (\,[[p]]obs_{ENV}(S)(t) = true) \Rightarrow ([[p]]obs_{RW}(t) = true)$$

# Confidence

S will use measures such as authorization, sensor fusion, fault tolerance, intrusion detection to ensure that currently relevant observation predicates are approximating reality with sufficient level of precision

We allow beliefs to be labeled by the (informal) notion of confidence levels

$$cl(S, \mathcal{E}(S)): R+ \rightarrow Beliefs(S) \rightarrow [0,1].$$

*Thus*

- *if* $p \in Beliefs(S)$ *depends on* $v_1,\ldots,v_n$ *and*
- $\mathcal{E}(S)(v_j)(t) = \in_j$
- *then* $cl(S, \mathcal{E}(S))(t)(p) = c$

  *indicates that system S will trust its belief p with confidence level c, assuming that the measurement errors of variables* $v_j$ *are bound by* $\in_j$

# An overarching design requirement

$$p \quad Beliefs(S)(t) \text{ and } cl(S, \mathcal{E}(S))(t)(p)=1$$

*iff p is true at time t in the real world with robustness* $\mathcal{E}(S)$

*“whenever my confidence in a belief is extremely high, than my belief coincides with reality up to robustness”*

# Environment models I

In order to predict the future evolution of the system, the system not only maintains beliefs about the current state of its environment, but also about the prevailing dynamics, such as

- its beliefs about vehicle dynamics model
- its belief about dynamics of weather conditions

# Environment models II

We distinguish between

- the actual real-world environment model, which determines for each System S and each of its extension variables their evolution taken into account the local variables of S controlling its own dynamics as well as disturbances given as a parametric probabilistic Hybrid Automata ENV(S)
- what S believes to be true about ENV(S), which we reduce to
  - the current mode of ENV(S)
  - the current valuation of its parameters

The system maintains PHAs predicting the expected dynamics of all relevant neighboring systems based on its beliefs about their class.

# Health state I

A probabilistic hybrid automaton $HA_H(S)$ defines the *health state* of *S* as follows:

$$HA_H(S) = (M_H(S), V_H(S), F_H(S), P_H(S), R_H(S), DI_H(S), init_H(S), inv_H(S), m_{0,H}(S))$$

where

- $M_H(S)$ is a finite set of *degradation modes of s, with mode invariants given by a labeling function* $inv_H(S)$;
- $V_H(S)$ is a set of *hazardous environment variables* (such as to model external physical forces acting on *S* causing its (partial) destruction, e.g. through collision) and local signals of *S* potentially influencing its failure behavior (such as its temperature), with initial valuation given by the predicate $init_H(S)$;
- A finite set $F_H(S)$ of *failure events* of *S* generated upon mode switches;
- A finite set of *parameters* $P_H(S)$;

# Health state II

- $R_H(S)$ defines with what probability and under what conditions on $V_H(S)$, failure events will be generated and a mode switch caused;
- $DI_H(S)$ specifies for each degraded modes *models for failure generation* through parameterized differential systems of equations (e.g. describing failure generation in harsh environments possibly including aging);
- $m_{0,H}(S)$ is the mode for *nominal* behavior of the system (where no failures have occurred, or after repair of all faulty components).

# Health state III

It is the task of system-level hazard analysis to identify hazardous environment signals.

System design must assure that they are *observable*, i.e. all hazardous environment signals must be contained in $V_{ENV}(S)$.

Similarly, system-level hazard analysis must identify all extensions of $S$ relevant for characterizing its failure behavior.

# Capabilities

... The prevailing environment dynamics of system S and its health state determine jointly the capabilities of the system, i.e. the set of behaviours which the system can potentially exhibit in this state.

Such capabilities may be restricted by roles.

A strategy will determine which of the available capabilities will be activated in order to achieve the goals of the system in that particular role.

# *Stabilization* Capabilities

… guarantee to maintain the state of the constituent system in a subspace described by some convex predicate  on its state space *V(S)*

- under assumptions specified in a contract
- with a given probability
- unless some exit condition  exit is triggered.

Thus such guarantees take the form

$$[\,]\ (\ \textit{unless}\ _{\textit{exit}}).$$

where the subscript  weakens the always operator, in that the formula *( unless $_{exit}$)* is now only expected to be true with probability .

# State-transition capabilities

… guarantee to transform

- the current (pre-) state $_{pre}$ of the constituent system to a (post-) state $_{post}$
- in a bounded time window with a given probability
- while maintaining a state invariant $_{inv}$ unless an exit condition $_{exit}$ is triggered.

Thus such guarantees take the form

$$[\,]((\,_{pre}\quad _{inv}\,)\quad _{inv}\,Until_{,}\,(\,_{post}\quad _{exit}\,)$$

where the double subscript $_{,}$ of the until operator states that a state meeting $(\,_{post}\quad _{exit}\,)$ is reached within time window with probability .

# Capabilities

We denote by $C(s, m_H, m_{env})$ the set of capabilities $S$ believes to be available in a given degradation mode and environment mode.

Each role $r$ defines its capabilities by picking a subset of these, which are then available to meet its goals, denoted by $C(S, r, m_H, m_{env})$.

Each role may impose global weak and strong assumptions, denoted by $assm_w(r)$ and $assm_s(r)$, respectively.

# Goals

Each role is equipped with a prioritized list of goals it is to achieve with the given capabilities.

Goals are formalized in timed probabilistic first-order LTL over the observables of a system.

# Strategies I

A strategy determines for a finite time window

> (called the *time-horizon of the strategy*)

how the system will react to

> changes of its interface variables (sensors, communication events)

> and its local state

> by the activation and deactivation of capabilities available in the current role, its current health state, and its current environment mode in order to achieve its current goals

# Strategies II

Defined as finite directed labelled trees, whose edges are labelled alternately along each path with activation/deactivation commands of available services, and valuation of its interface variables.

The nodes of the tree are labelled with the belief about the visible state of the system when responding to the sequence of observations about its environment along the path leading to this node with the sequence of activation/deactivation of its services leading to this node.

The root of such a decision tree is by definition labelled with the current state.

Recall that the visible state includes the beliefs of the state of the environment, its own local state, and the state of systems it owns or knows.

# Winning strategies

A strategy is a *winning strategy* in time horizon Δ if it achieves all its current safety goals and all its time-bounded reachability properties expiring in Δ.

We replace exact satisfaction of LTL formula by robust satisfaction, where small perturbations of the model are not allowed to cause valid formula to become invalid.

# More on strategies

The time horizon of a strategy will be typically chosen taking into account the assumed environment model and the short-term goals.

To determine such a strategy, a system will use its belief about the environment model to assess (approximately) the future evolution of the real-world state based on its currently believed state up to the time horizon of the strategy, e.g. using tools for robust reachability analysis of non-linear hybrid systems.

If this analysis shows that no winning strategy exists, then the goals that are violated are flagged as *unachievable*.

# Cooperative Systems I

*If S* finds that it alone can not achieve its goals, it might choose to send cooperation requests to neighboring systems.

Formally, if another system accepts a cooperation request, it adds the current goals of the system requesting help to its own list of goals, thus adapting its own current strategy to also take into account these new goals.

# Cooperative Systems II

Typically, these systems would also exchange their respective beliefs about the real world and agree to a shared view using *belief fusion* (a generalization of *sensor fusion*).

Intuitively, belief fusion resolves inconsistent beliefs based on confidence levels, and simply extends the beliefs with beliefs about objects not previously observed by the other system. This includes in particular beliefs about the prevailing environment dynamics.

Strategy synthesis in cooperating system is thus carried out based on consistent beliefs about the environment.

# What if a winning strategy does not exist

When a winning strategy does not exist in a particular situation – such as when cooperation requests are declined -, it is up to the supervisor to either change the role of *S* to one with more capabilities or to allow *S* to *use* other systems.

Using other systems will allow *S* to activate/deactivate the capabilities of these system as if they were its own. Thus, the capabilities of *S* are extended with the capabilities of systems it is allowed to use.

# A strategy believed to be winning need not be winning in reality:

- The synthesis of the strategy is by necessity based on the system´s beliefs about the environment and itself.
- If its beliefs are poor, than following the strategy will lead to situations where the actually observed state at some point in time t will differ from the state the strategy expected to reach at that point in time.
- To be able to make these assessments, we have included the expected state to be reached as node labels.

# What if …

- If the prediction about the future system state turns out to be incorrect, the execution of the strategy must be abandoned, and a new strategy must be synthesized based on the updated beliefs.
- This learning step will typically also involve updating beliefs about parameters of the environment mode.
- This can be accomplished by comparing sequences of
  - actually observed sensor data
  - with the expected beliefs, based on the internal representation of the environment dynamics in the current environment mode,
  - or even learning about mode-switches in the environment model when parameter-fitting methods are not able to explain the deviations between expected and actually observed trajectories.

# Conclusion

We provided a formal specification of an overarching correctness requirement on the perception chain.

We presented a formal semantics based on games in dynamically changing interaction structures between hierarchically organized systems, whose behavior can be captured through probabilistic hybrid automata.

Conflicts between local objectives and global objectives become explicit by non-existence of winning strategies.

Conflict resolution strategies are made explicit, e.g. role changes, cooperation requests, delegating additional resources, etc.

# LINKS AND REFERENCES

# Relevant Roadmaps Germany/EU

National Roadmap on Embedded Systems

Agenda CPS, acatech

New autoMobility – The Future World of Automated Road Traffic, acatec

Drafts MASRIA Joint Undertaking ECSEL

SRA ETP Artemis

Automotive Roadmap Embedded Systems 2030

Findings of the SafeTRANS Working Group on Highly Automated Systems

PROPOSAL OF A EUROPEAN RESEARCH AND INNOVATION AGENDA ON CYBER-PHYSICAL SYSTEMS OF SYSTEMS 2016-2025

Round Table Autonomous Driving

Industrie 4.0

# References I


Werner Damm and Alberto Sangiovanni-Vincentelli, *"A Conceptual Model of Systems of Systems"*, Proc Second International Workshop on the Swarm at the Edge of the Cloud at CPS Week 2015, April 2015, Seattle

Werner Damm and Roland Galbas, "*Exploiting Learning and Scenario-based Specification Languages for the Verification and Validation of Highly Automated Driving"* Proc SEFAIAS'2018, 28.05.2018, Gothenburg, Sweden

Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto L. Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger, Kim G. Larsen: *"Contracts for System Design"*. Foundations and Trends in Electronic Design Automation 12(2-3): 124-400 (2018)

Werner Damm, Martin Fränzle, Sebastian Gerwinn, Paul Kröger *Perspectives on the Validation and Verification of Machine Learning Systems in the Context of Highly Automated Vehicles,* SIRLE 2018: AAAI 2018 Spring Symposium on Integrating Representation, Reasoning, Learning, and Execution for Goal Directed Autonomy, Palo Alto

Eckard Böde, Matthias Büker, Werner Damm, Günter Ehmen, Martin Fränzle, Sebastian Gerwinn, Thomas Goodfellow, Kim Grüttner, Bernhard Josko, Björn Koopmann, Thomas Peikenkamp, Frank Poppen, Philipp Reinkemeier, Michael Siegel, and Ingo Stierand. *"Design Paradigms for Multi-Layer Time Coherency in ADAS and Automated Driving (MULTIC)"*. FAT-Schriftenreihe. Forschungsvereinigung Automobiltechnik e.V. (FAT), 302 edition, 2017.

# References II


Werner Damm, Martin Fränzle, Sebastian Gerwinn, Paul Kröger *Perspectives on the Validation and Verification of Machine Learning Systems in the Context of Highly Automated Vehicles,* SIRLE 2018: AAAI 2018 Spring Symposium on Integrating Representation, Reasoning, Learning, and Execution for Goal Directed Autonomy, Palo Alto, CA, 2018

Werner Damm, Peter Heidl: SafeTRANS Working Group *"Highly automated Systems: Test, Safety, and Development Processes",* Recommendations on Actions and Research Challenges, 2017, available from http://www.safetrans-de.org/de/Aktuelles/positionspapier-zu-hochautomatisierten-systemen/2

Werner Damm, Eike Möhlmann, Thomas Peikenkamp and Astrid Rakow, *"A Formal Semantics for Traffic Sequence Charts",* Booktitle: Festschrift in honor of Edmund A. Lee, October 2017, to appear in Springer Lecture Notes in Computer Science, 2018

W. Damm, S. Kemper, E. Möhlmann, T. Peikenkamp and A. Rakow, *"Traffic Sequence Charts - A Visual Language for Capturing Traffic Scenarios",* Embedded Real Time Software and Systems - ERTS2018, February 2018

# References III


Werner Damm, Bernd Finkbeiner and Astrid Rakow *"What You Really Need To Know About Your Neighbor",* Proceedings at 5th Workshop on Synthesis (SYNT 2016), Toronto, Ontario, Canada, July 2016.

Werner Damm, Bernd Finkbeiner, "*Automatic Compositional Synthesis of Distributed Systems*", accepted for publication at the 19th International Symposium on Formal Methods, Singapur, May 2014.

Werner Damm, Eike Möhlmann, Astrid Rakow, *Component Based Design of Hybrid Systems: A Case Study on Concurrency and Coupling*, 17th International Conference on Hybrid Systems: Computation and Control (HSCC 2014), Berlin, April 2014.

W. Damm, H.-J. Peter, J. Rakow, and B. Westphal, "*Can we build it: formal synthesis of control strategies for cooperative driver assistance systems,*" Mathematical Structures in Computer Science, vol. 23, iss. 4, pp. 676 - 725, 2013.

Werner Damm and Bernd Finkbeiner. *Does it pay to extend the perimeter of a world model?* In Michael Butler and Wolfram Schulte, editors, Proceedings of the 17th International Symposium on Formal Methods, Lecture Notes in Computer Science, pages 12--26, June 2011