

Spatio-Temporal Stream Reasoning for Safe Autonomous Systems

Fredrik Heintz, Dept. of Computer Science,
Linköping University, Sweden

fredrik.heintz@liu.se

@FredrikHeintz

<http://www.ida.liu.se/~frehe08/streamreasoning/>

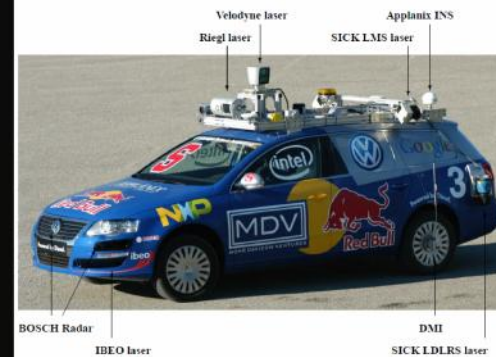


Outline

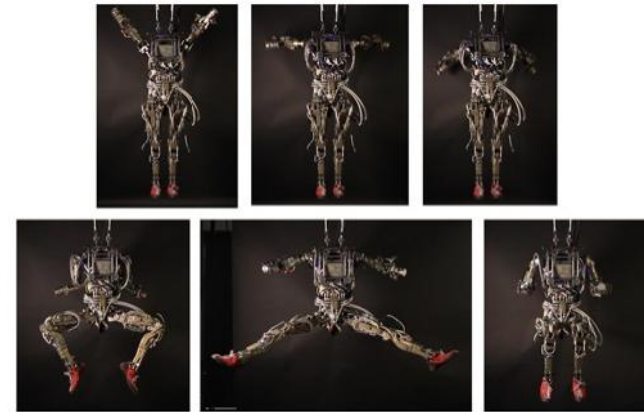
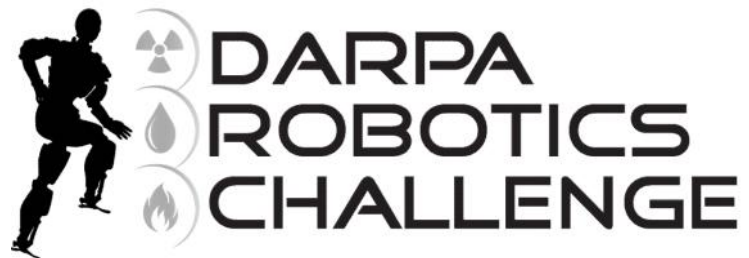
- Introduction to Autonomous Systems as an application area for stream reasoning
- It's a Streaming World! Other Stream Reasoning Applications
- Related work: Data Stream Management Systems, Complex Event Processing and Stream Reasoning for the Semantic Web
- Spatio-Temporal Logic-Based Stream Reasoning using Metric Temporal Logic through Progression over complete information
- Spatio-Temporal Logic-Based Stream Reasoning with Incomplete Information
- Execution Monitoring through Stream Reasoning for Safe Autonomous Systems
- Grounding Logic-Based Stream Reasoning for Autonomous Systems
- Future Work: Probabilistic and Anticipatory Stream Reasoning
- Summary and Conclusions

Autonomous Systems and Stream Reasoning

Autonomous Systems



Autonomous Systems



Collaborative Unmanned Aircraft Systems



A principled approach to building collaborative intelligent autonomous systems for complex missions.

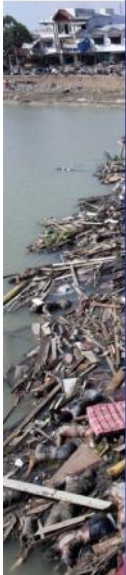


Vision

A principled approach to building collaborative intelligent autonomous systems for complex missions.

Challenges:

- Support humans and robots including legacy systems
- Support adjustable autonomy and mixed-initiative interaction
- Manage tasks and information on many abstraction levels
- Coordinating control, reaction and deliberation
- Coordination of systems, resources and platforms
- Incomplete information at design time and run time
- Inspection, monitoring, diagnosis and recovery on many abstraction levels



 SHERPA

Patrolling hawks

Trained wasps

Autonomous Systems at AIICS, Linköping University



Micro UAVs
weight < 500 g,
diameter < 50 cm



Yamaha RMAX
weight 95 kg,
length 3.6 m



LinkQuad weight ~1 kg, diameter ~70cm

RMAX System Overview



- 3xPC104 computers
- 802.11 wireless bridge
- GPS, barometric altimeter
- Color & thermal cameras (PTU)
- Video transmitter & recorder
- Laser range finder

- 21 HP two-stroke engine
- 3.6 meters length
- Maximum takeoff weight – 95 kg
- Radio controlled (backup pilot)
- Attitude sensor (YAS) & stabilization system (YACS)



SELECTED AUTONOMOUS FUNCTIONALITIES

LINKÖPING UNIVERSITY, SWEDEN

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE

AUTONOMOUS UNMANNED AERIAL VEHICLE TECHNOLOGIES LAB

LinkQuad



Dimensions

- Width: 68.5 cm
- Width (w/o props): 45.8 cm
- Top diameter: 68.5 cm
- Height: 20 cm
- Propeller size: 10" (25.4 cm)

Weight and Payload

- Empty weight: 950 g
- Payload capacity: 250 g
- Max take-off weight: 1250 g

Power System

- 4x brushless motors
- Custom designed ESC
- Battery modules (2.7Ah or 5.4Ah)
- Flight duration: up to 40 min.
depending on the configuration



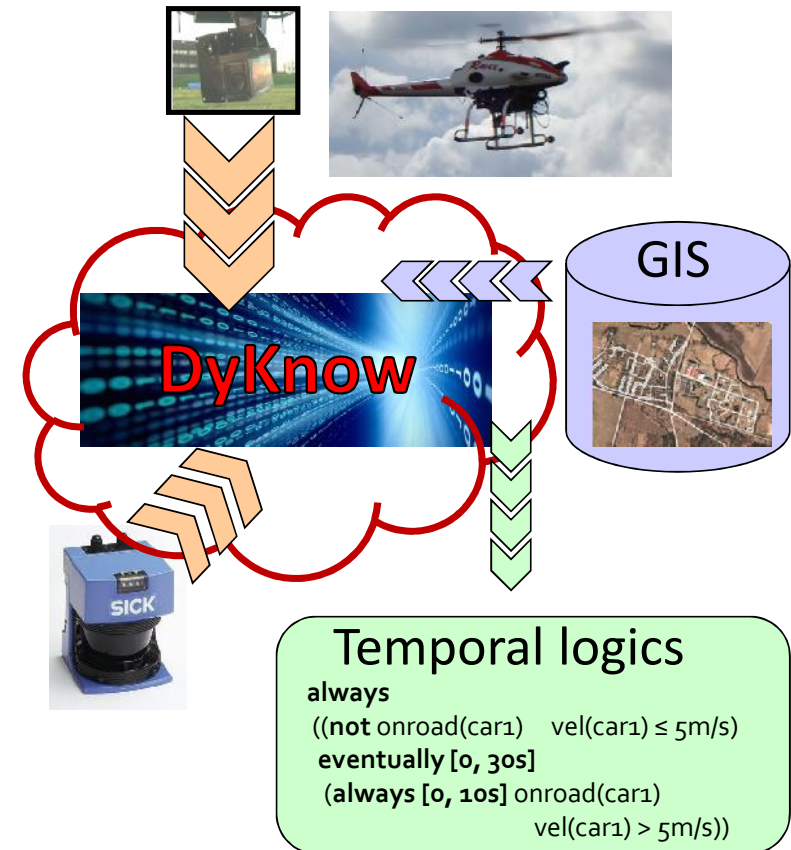
Battery module



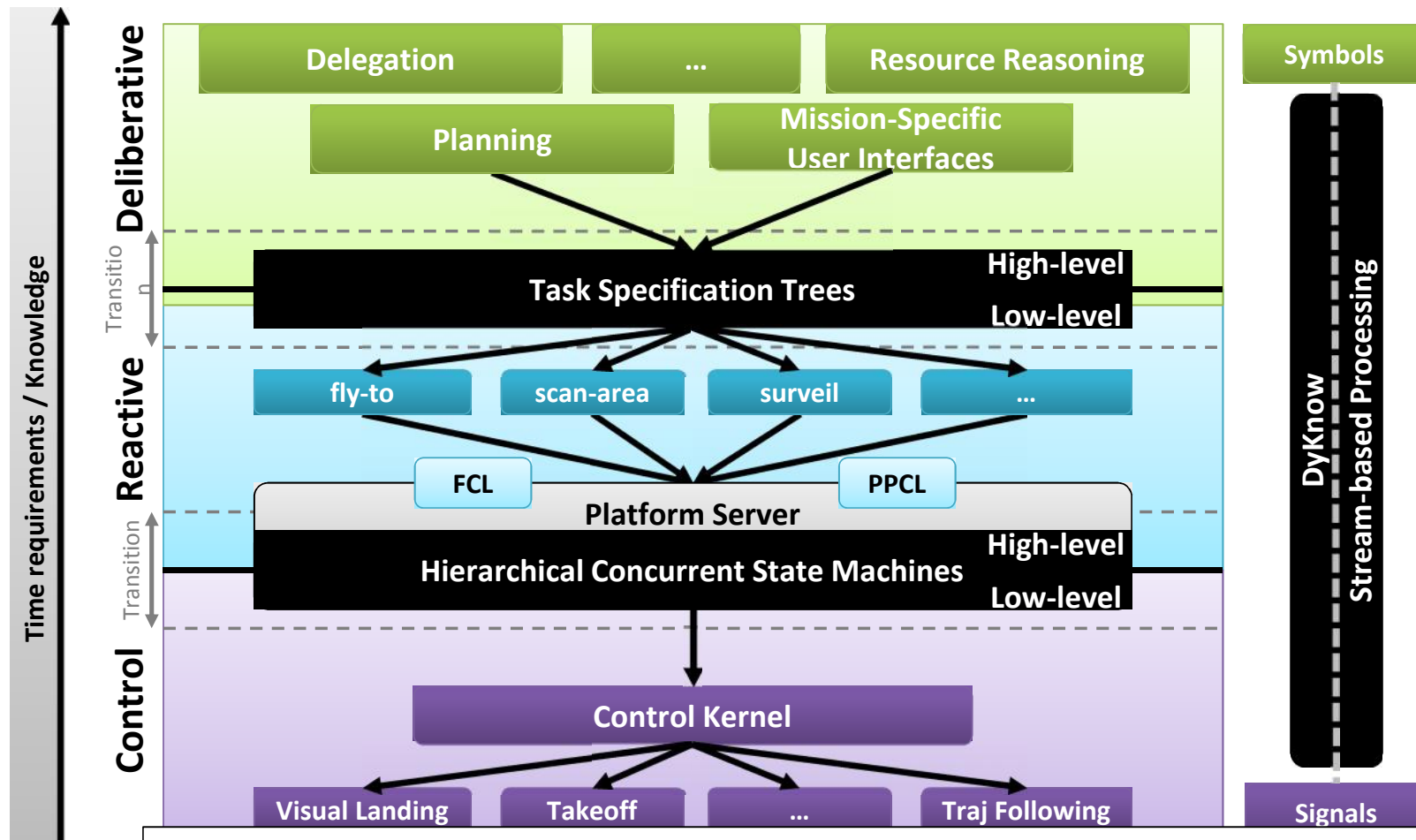
Twist and lock sensor modules

Stream Reasoning

- Autonomous systems produce and process sequences of values incrementally created at run-time.
- These sequences are natural to model as *streams*.
- *Stream reasoning* is incremental reasoning over streams.
- Stream reasoning approximates continuous reasoning with minimal latency necessary in order to react in a timely manner to changes in the environment.



HDRC3: A Distributed Hybrid Deliberative/Reactive Architecture for Autonomous Systems

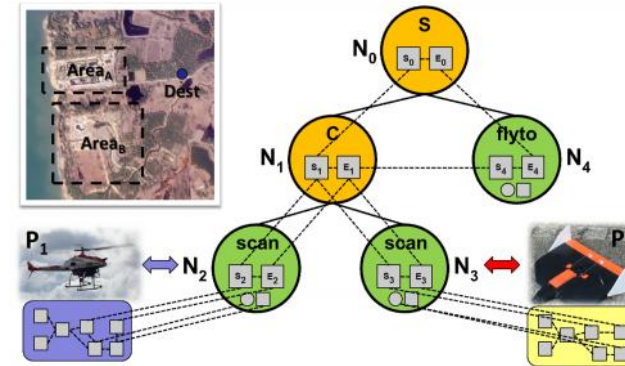
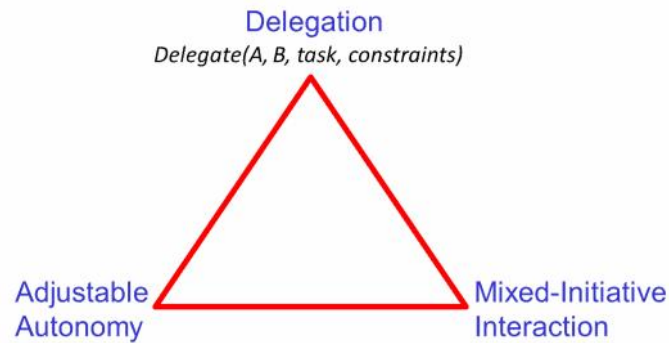


P. Doherty, J. Kvarnström, M. Wzorek, P. Rudol, F. Heintz and G. Conte. 2014.

HDRC3 - A Distributed Hybrid Deliberative/Reactive Architecture for Unmanned Aircraft Systems.

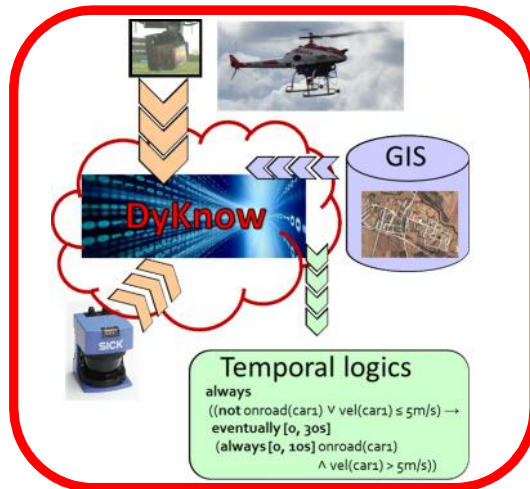
In K. Valavanis, G. Vachtsevanos, editors, Handbook of Unmanned Aerial Vehicles, pages 849–952.

My Contributions to HDRC3

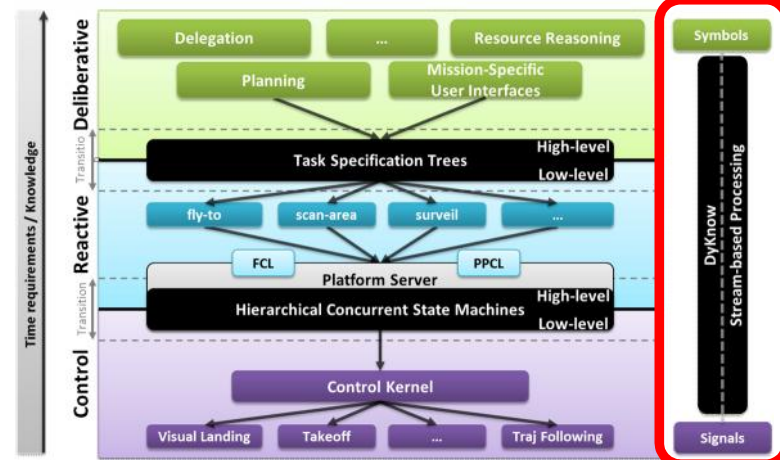


Delegation and task allocation through constraint satisfaction

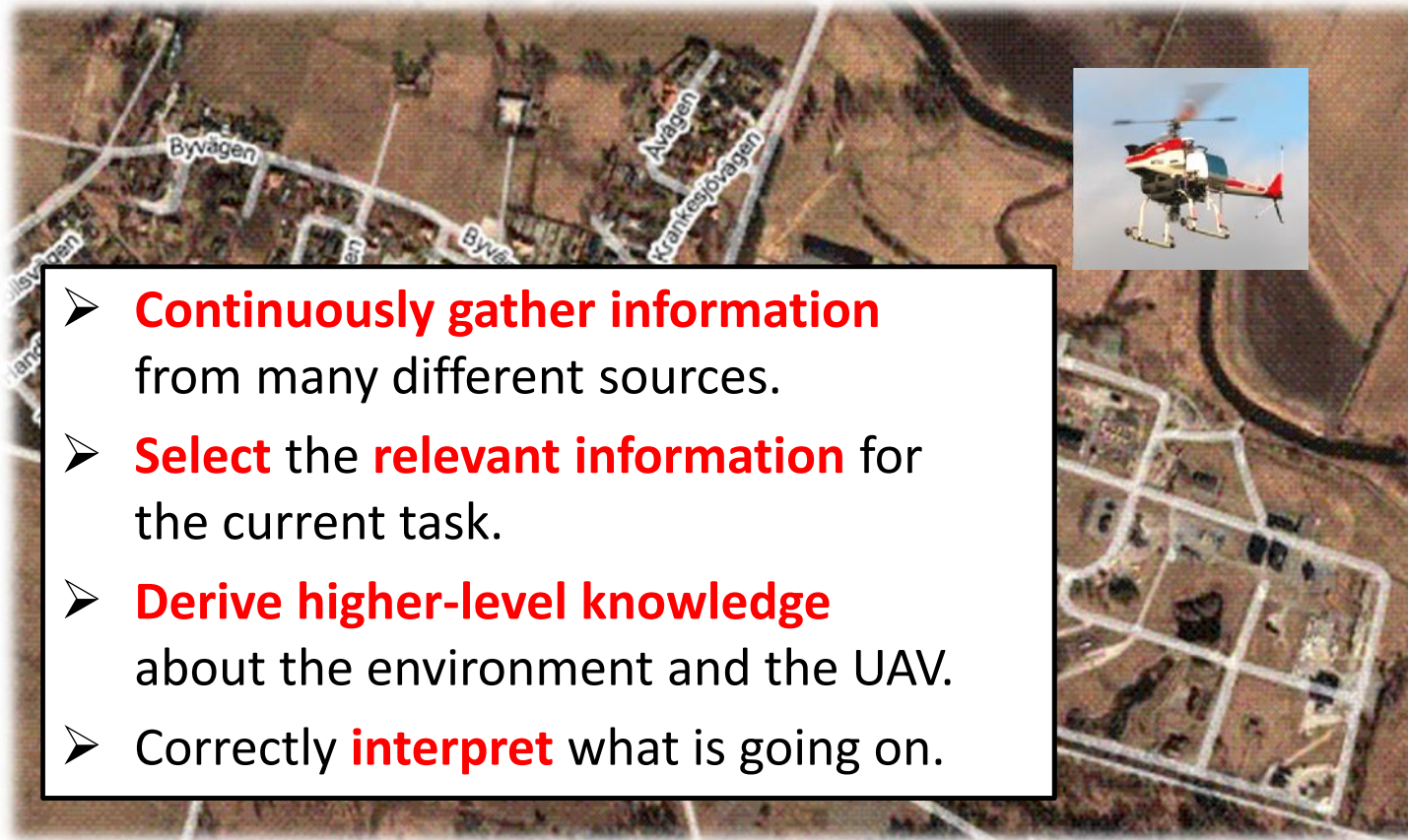
Stream reasoning



Architectures



A Traffic Monitoring Scenario

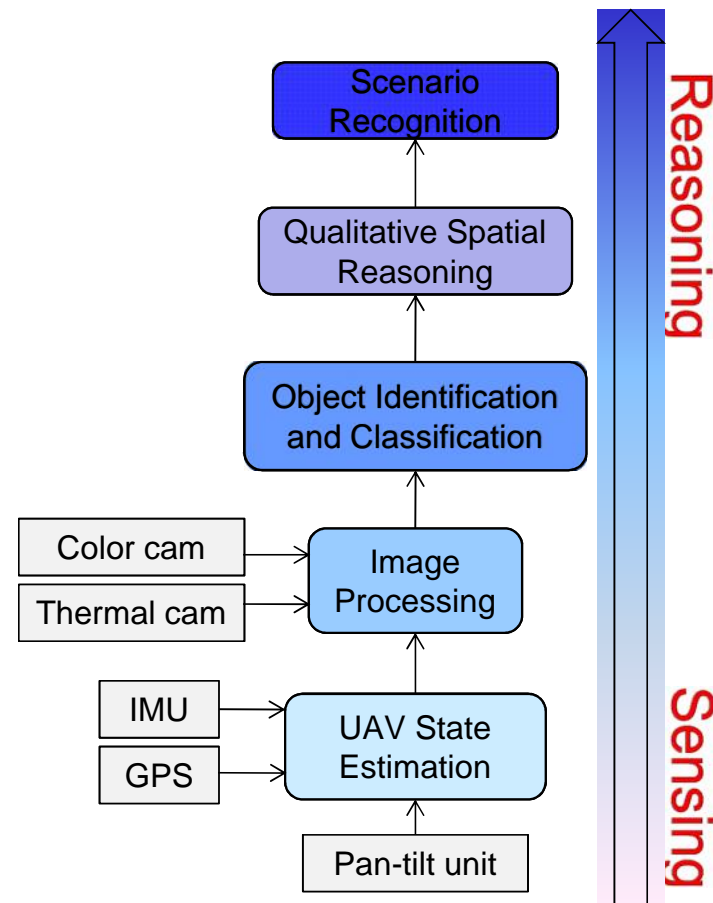


- **Continuously gather information** from many different sources.
- **Select** the **relevant information** for the current task.
- **Derive higher-level knowledge** about the environment and the UAV.
- Correctly **interpret** what is going on.

DyKnow

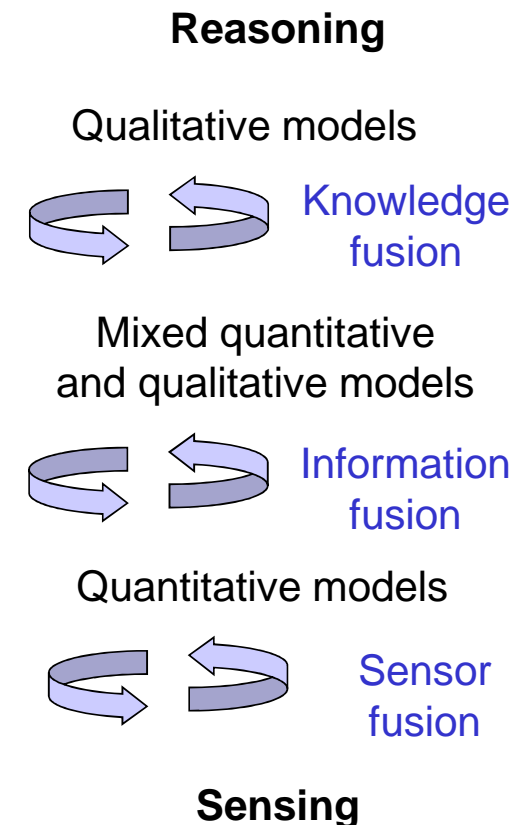
DyKnow is a stream-based knowledge processing middleware framework that provides

- a formal conceptual framework for integrating different sensing and reasoning approaches in a coherent processing framework,
- stream reasoning functionalities, and
- a distributed implementation infrastructure.

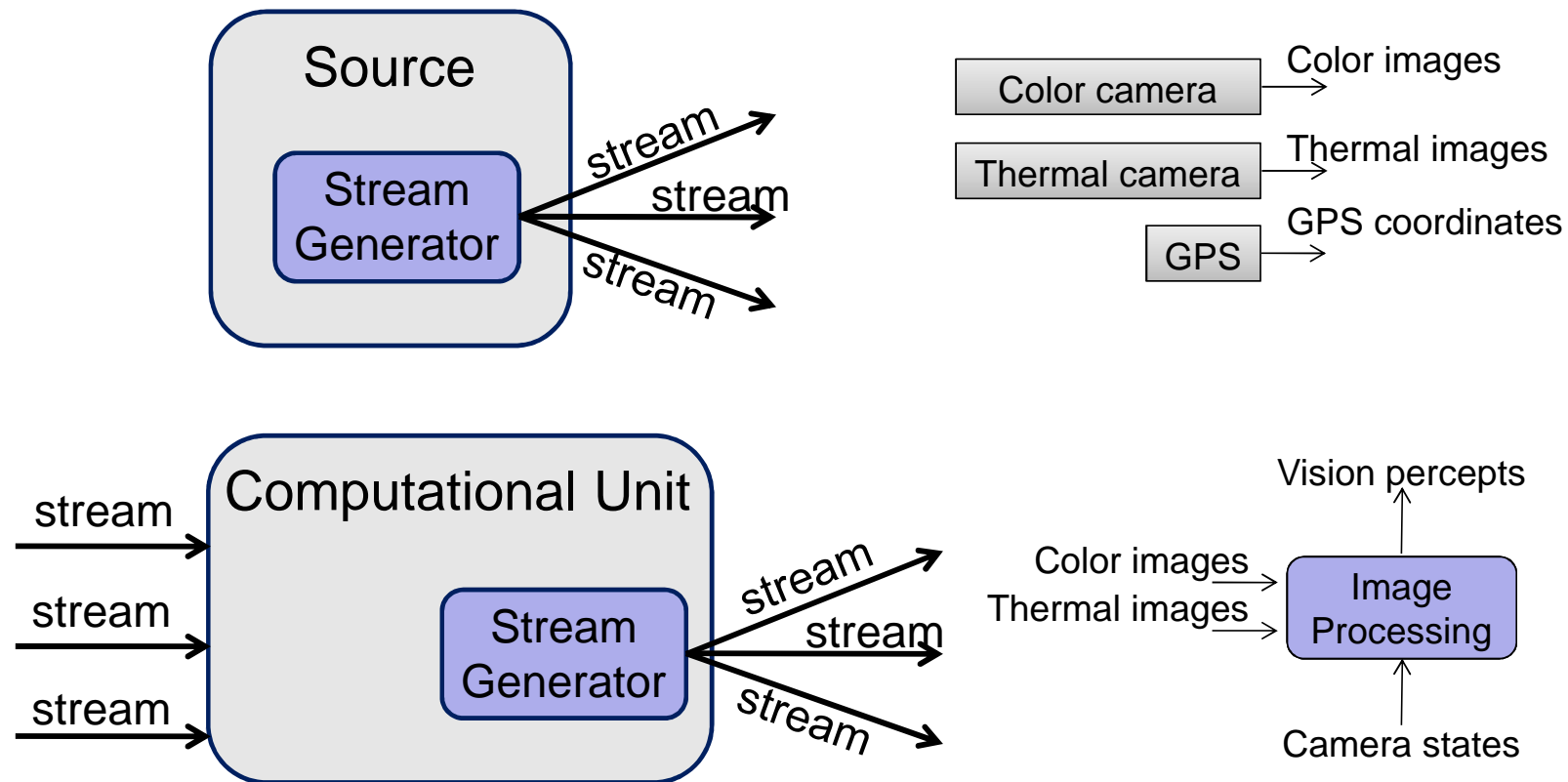


DyKnow: Requirements

- Integrating information from distributed sources.
- Processing at many different levels of abstraction.
- Quantitative and qualitative processing.
- Bottom-up data processing and top-down model-based processing.
- Managing uncertainty on different levels of abstraction.
- Flexible configuration and reconfiguration.
- Declarative specification of the information being generated and the available information processing functionalities.

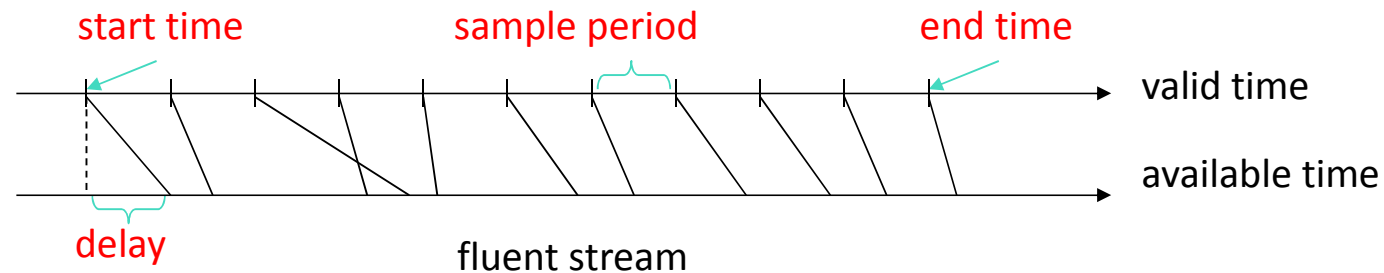


DyKnow: Sources and Computational Units



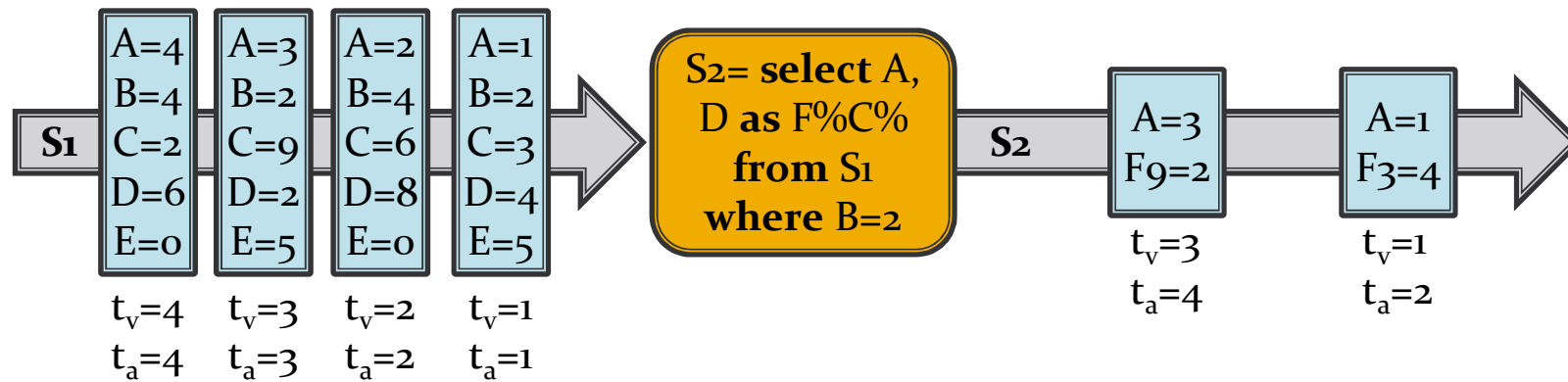
DyKnow: Policy

from 0 to 120s sample every 100ms max delay 200ms

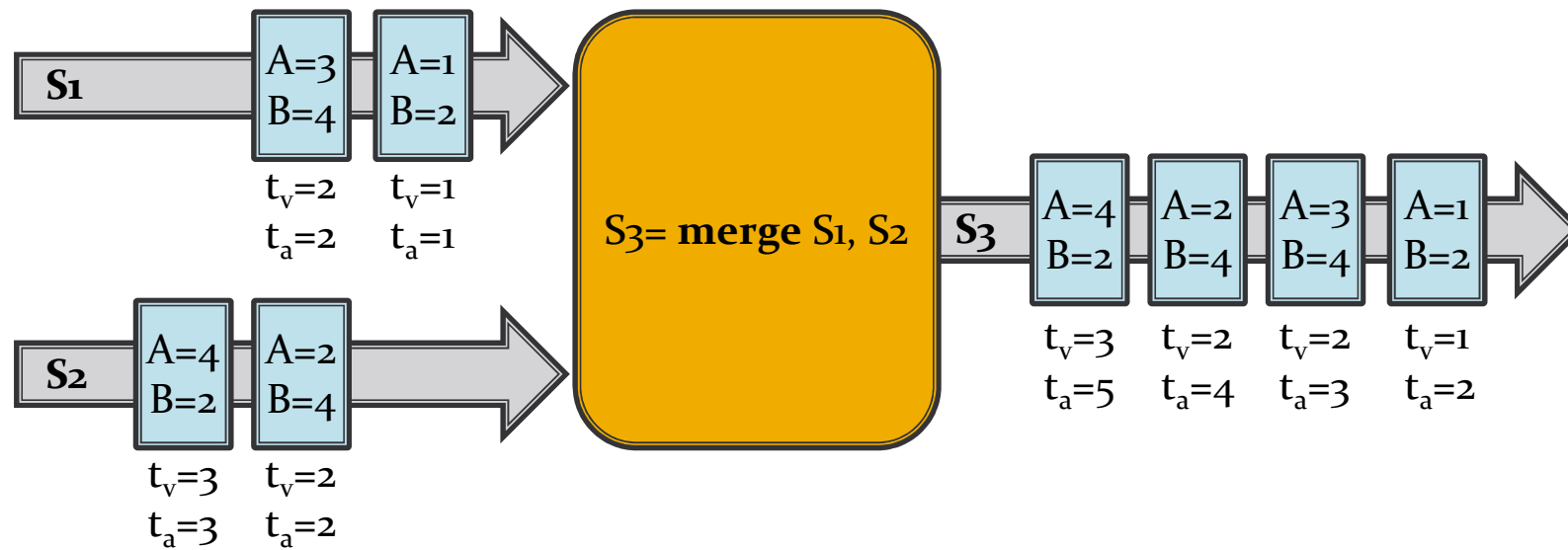


```
FLUENT_STREAM_POLICY := FLUENT_STREAM_CONSTRAINT*  
FLUENT_STREAM_CONSTRAINT :=  
    VALUE_APPROXIMATION_CONSTRAINT  
    | CHANGE_CONSTRAINT  
    | DELAY_CONSTRAINT  
    | DURATION_CONSTRAINT  
    | ORDER_CONSTRAINT
```

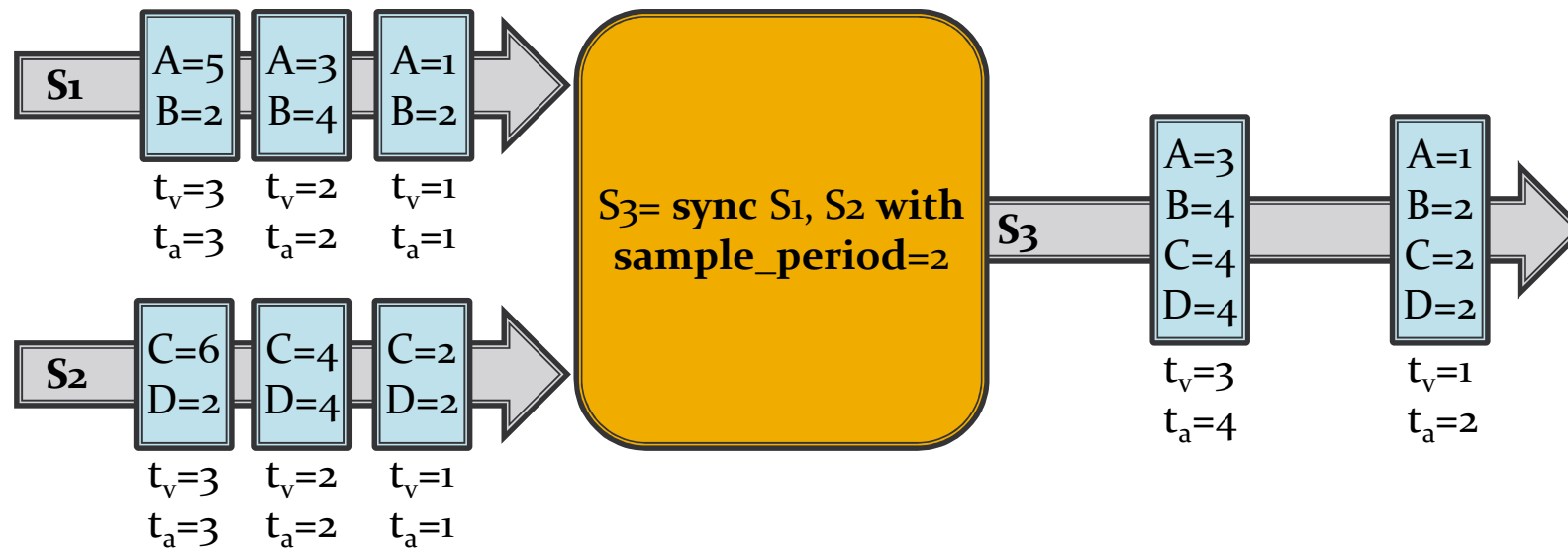
DyKnow: **Select**



DyKnow: Merge

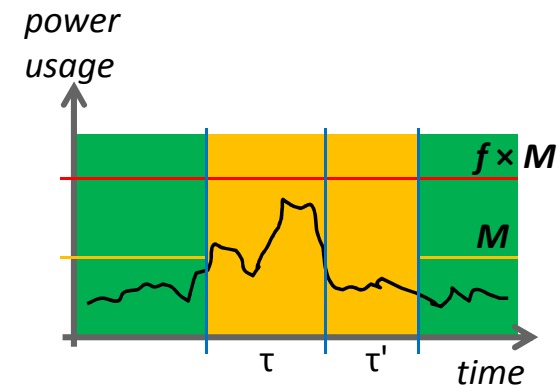


DyKnow: Sync

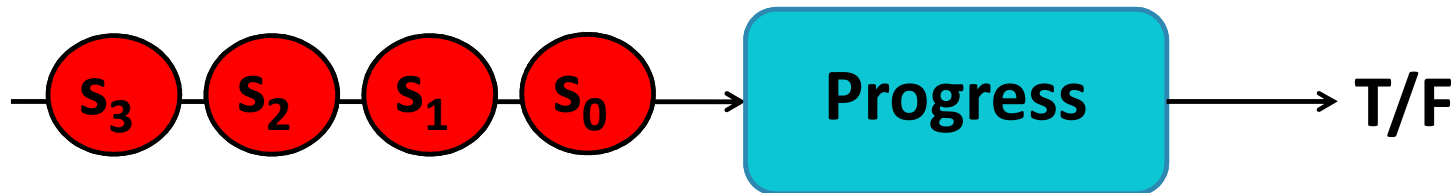


DyKnow: Logic-based Stream Reasoning

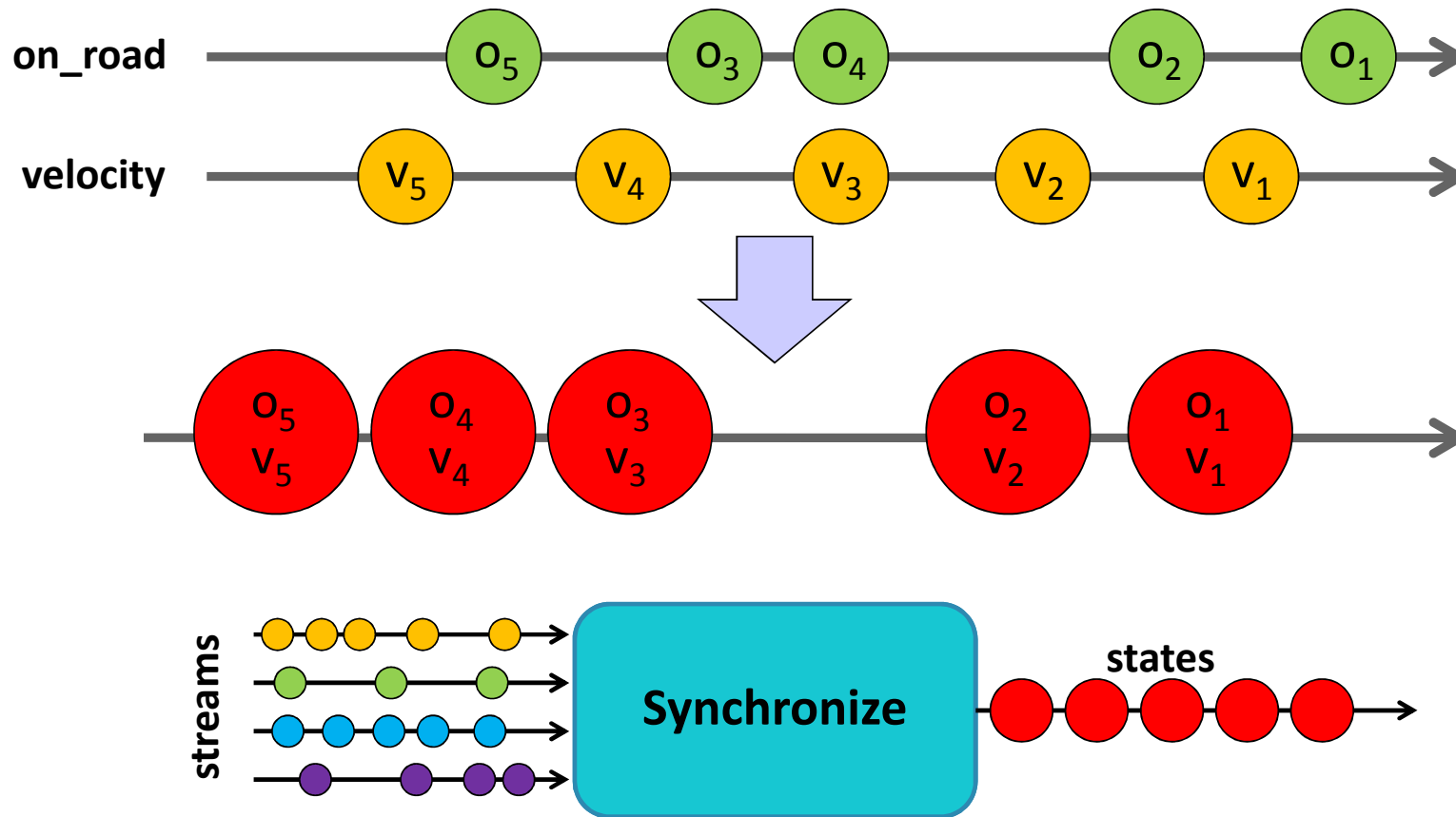
always $\forall uav. ((\text{power_usage}(uav) > M) \rightarrow$
 $((\text{power_usage}(uav) < f \times M)$
 until $[0, \tau]$
 $(\text{always}[0, \tau'] \text{ power_usage}(uav) \leq M)))$



The semantics of these formulas is defined over infinite state sequences. Progression is one technique to check whether the current prefix is sufficient to determine the truth value of a formula.



DyKnow: Generating State Streams

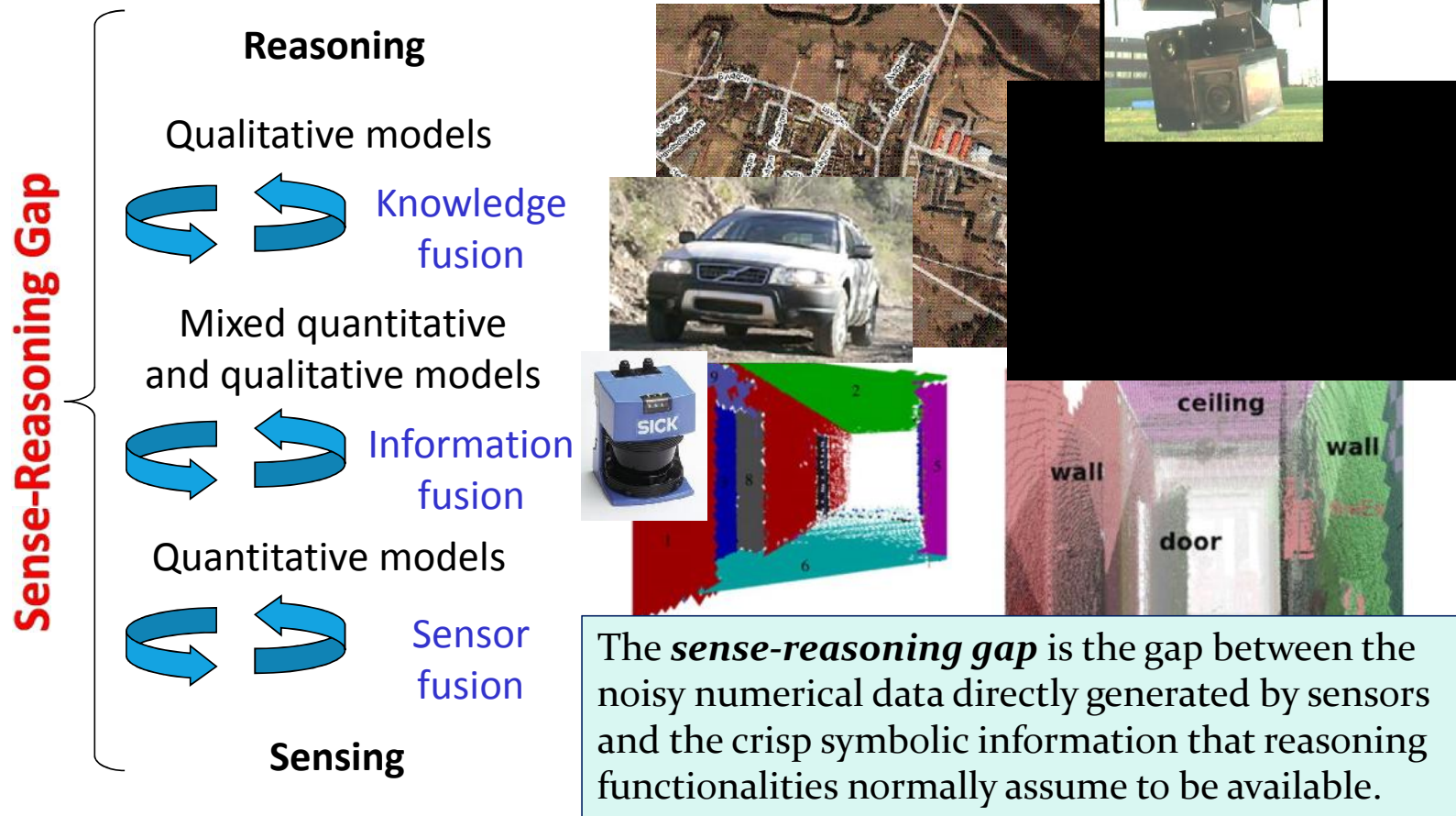


DyKnow: Chronicle Recognition

```
chronicle overtake[?car1, ?car2] {  
  event(behind[?car1, ?car2]:(F, T), t1)  
  event(beside[?car1, ?car2]:(F, T), t2)  
  event(in_front_of[?car1, ?car2]:(F, T), t3)  
  t1 < t2  
  t2 < t3  
  t3-t1 in [0, 30s]  
}
```

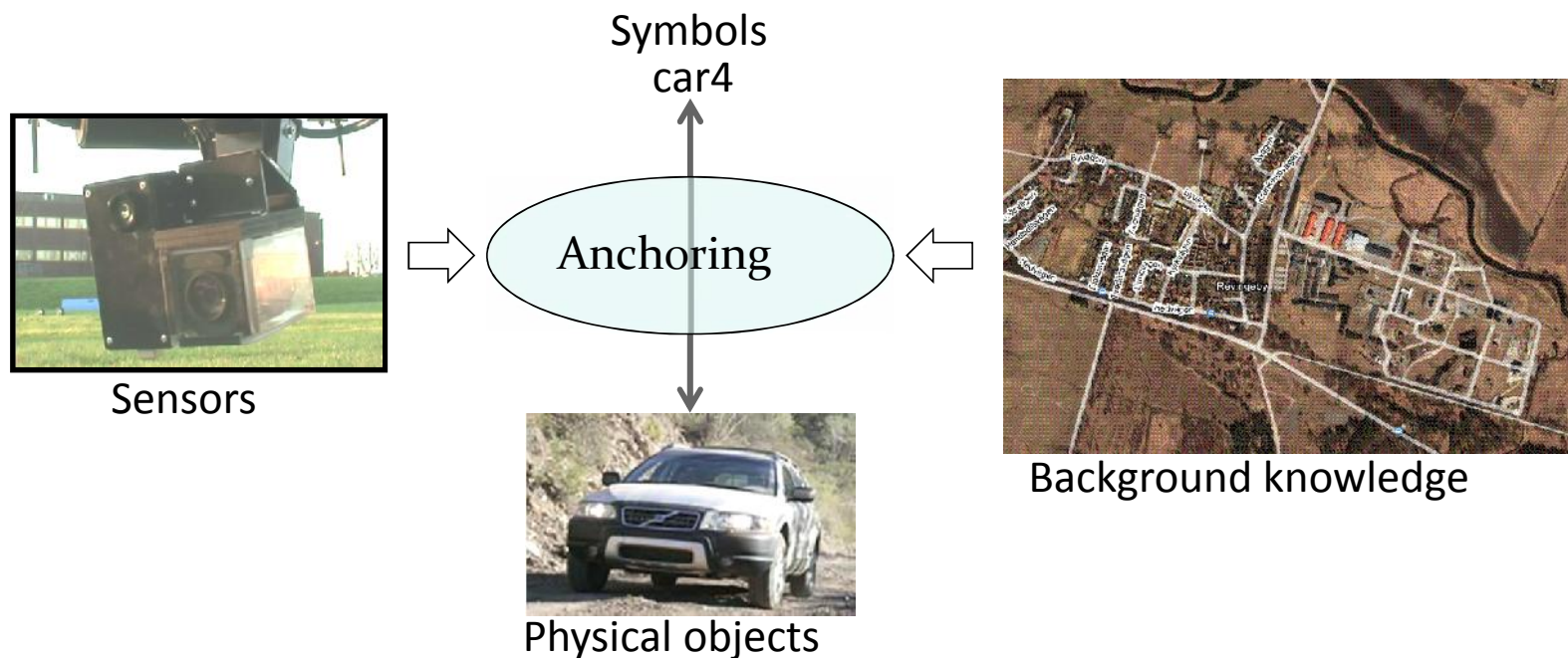


The Sense-Reasoning Gap

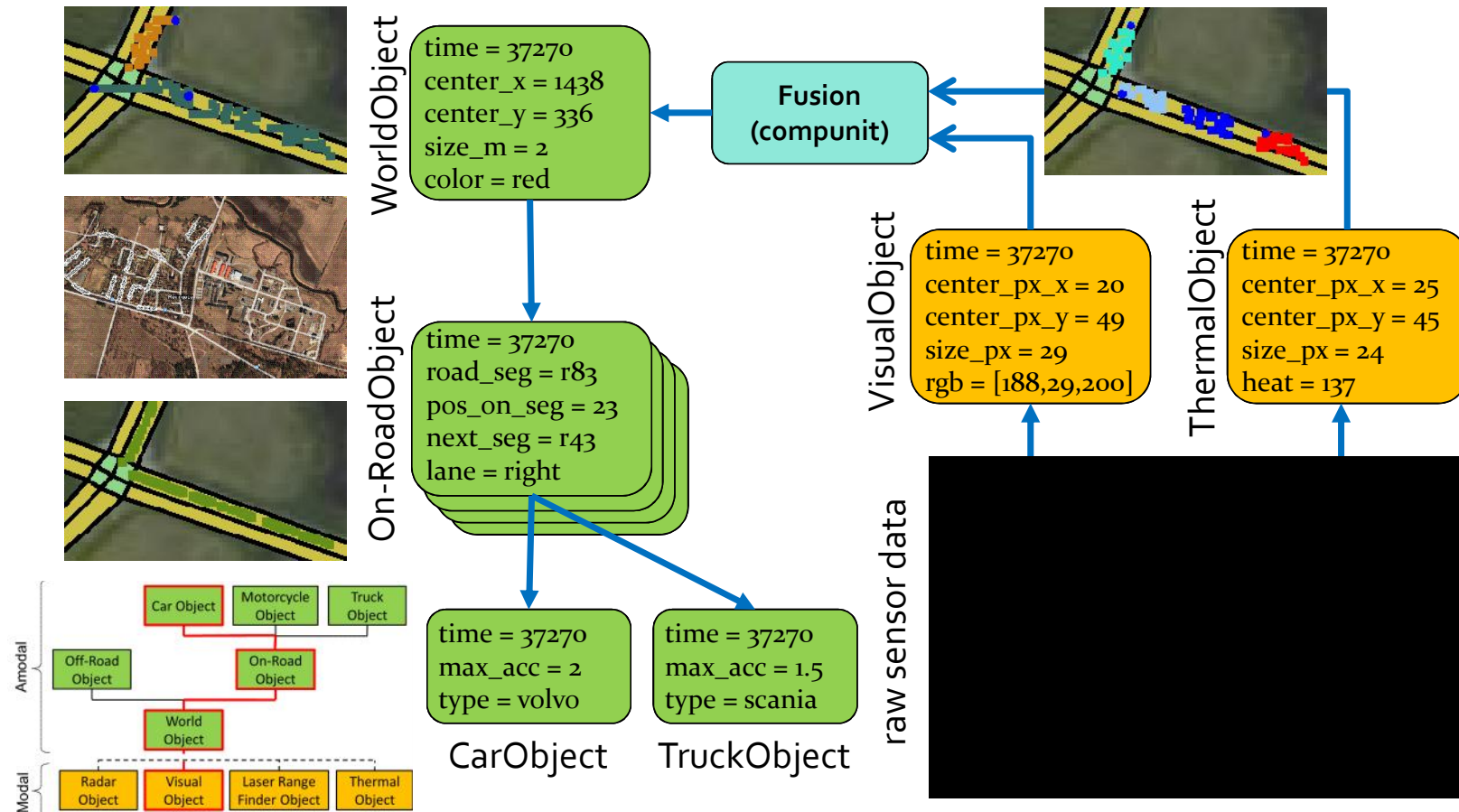


DyKnow: **Anchoring**

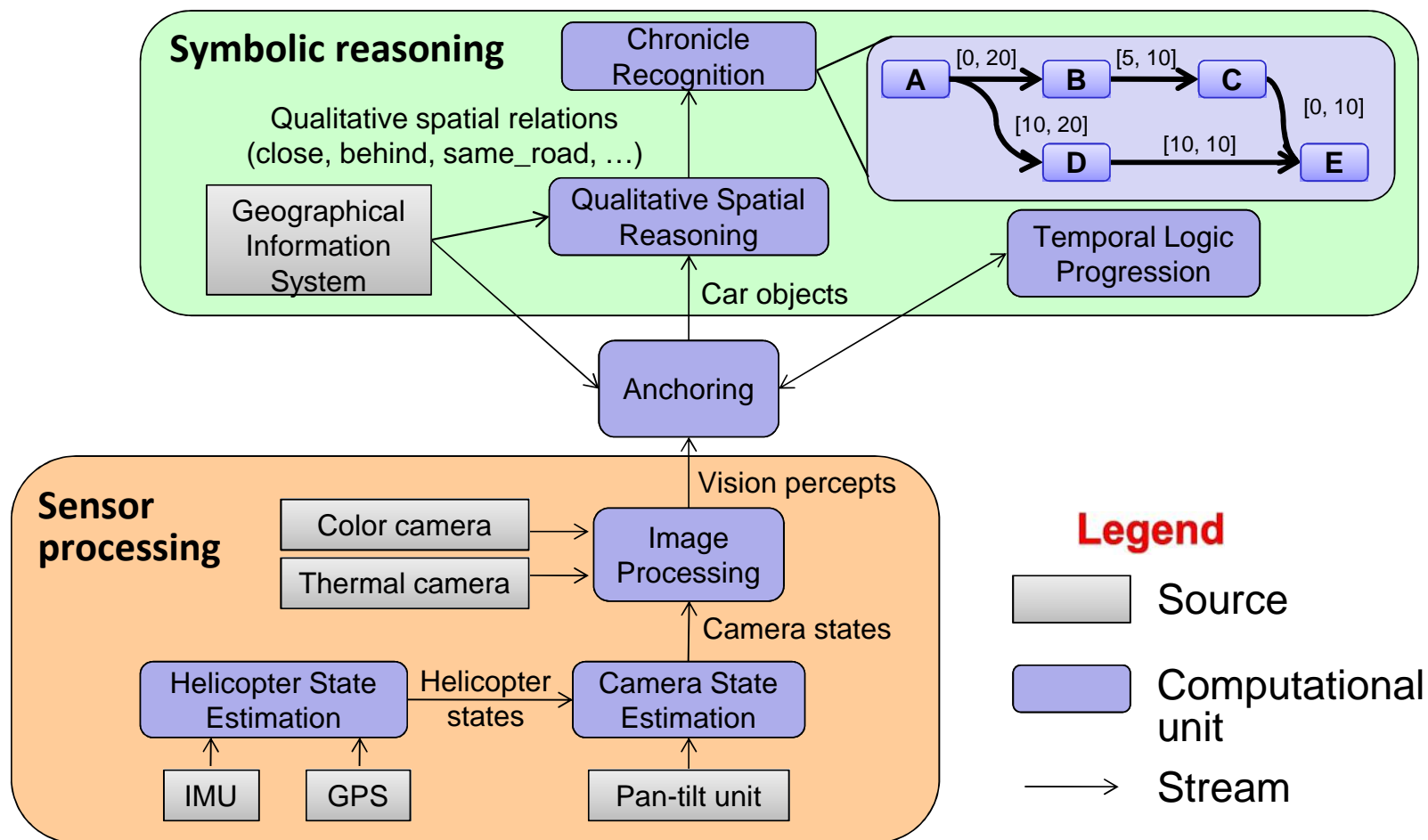
The objective of the anchoring process is to connect symbols to sensor data originating in the physical world so that the symbols represent the objects in the world.



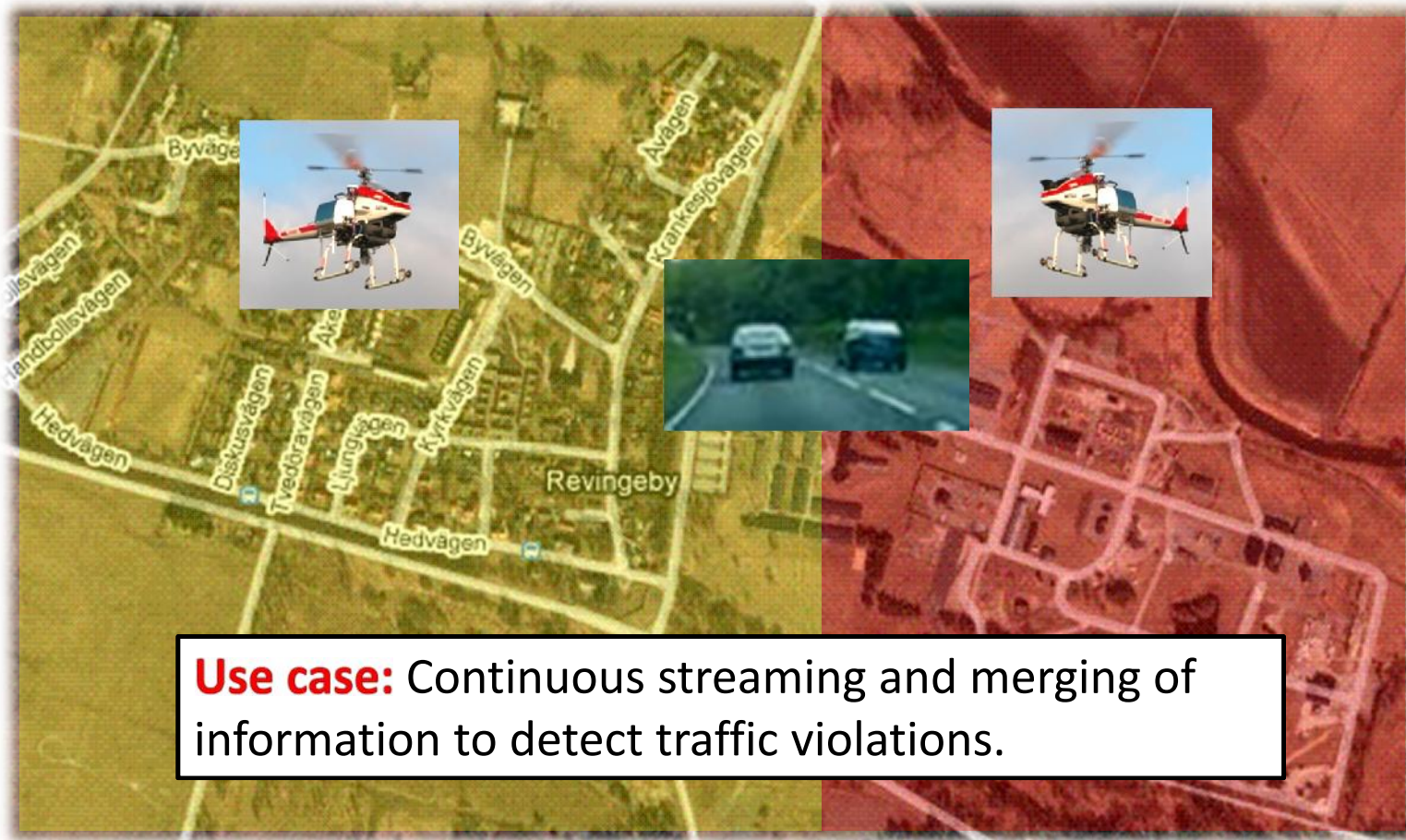
DyKnow: Bridging the Sense-Reasoning Gap



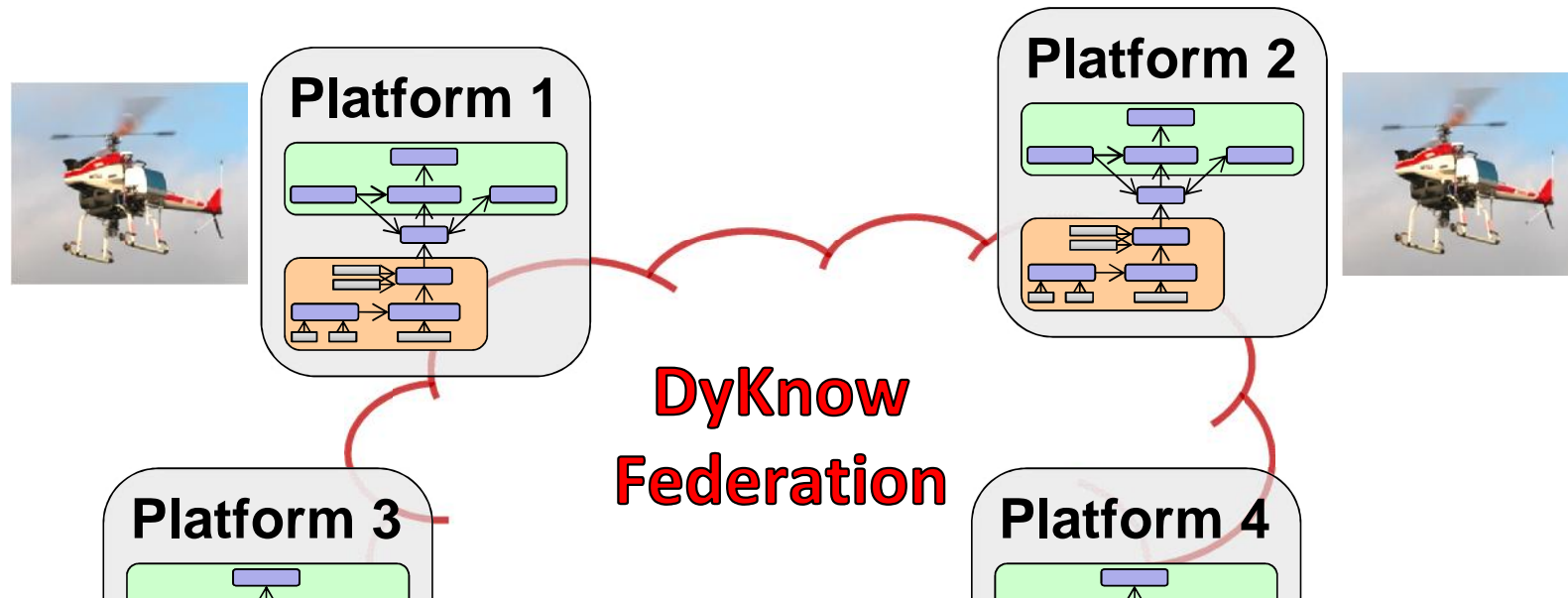
DyKnow Application: **Traffic Monitoring**



Multi UAV Traffic Monitoring

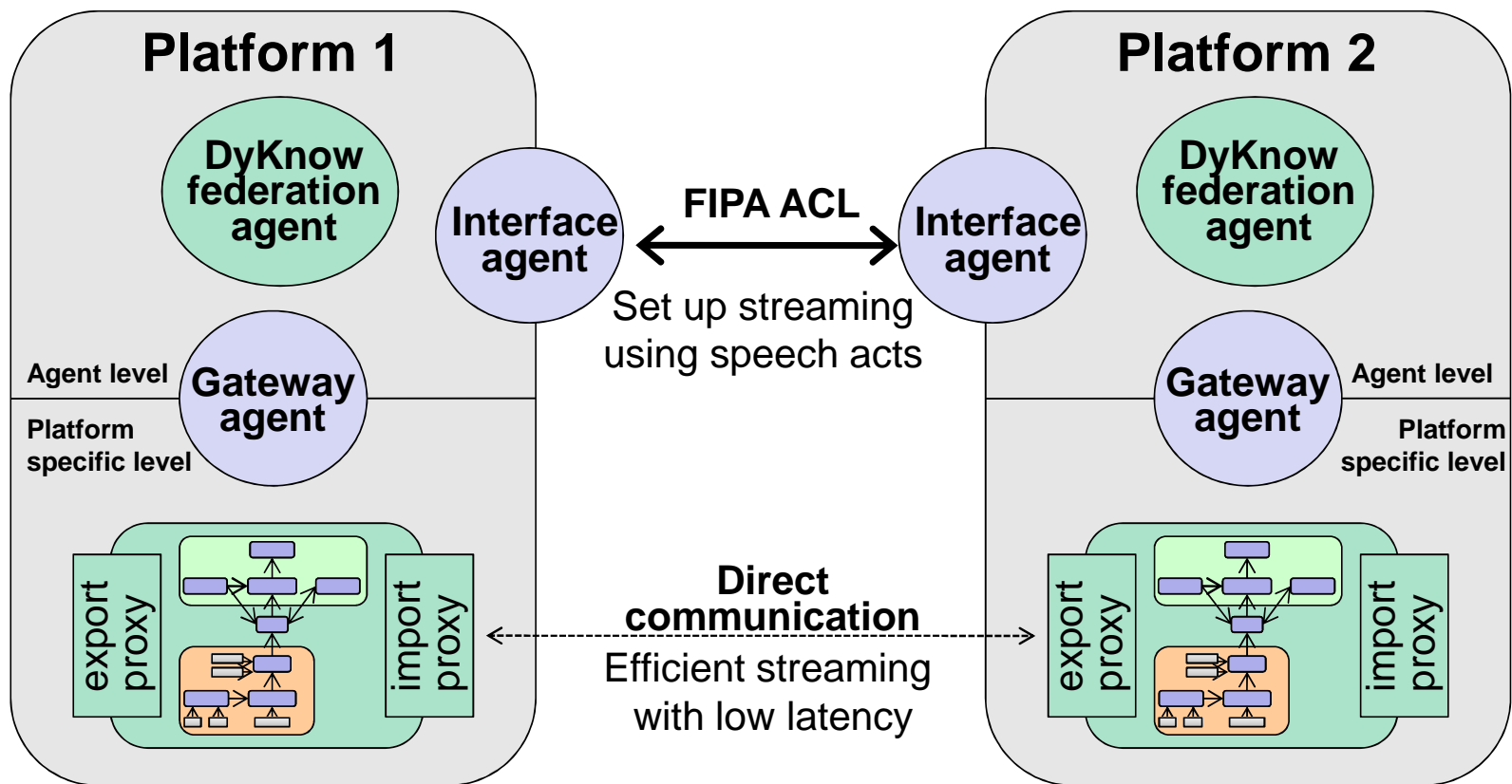


Finding and Sharing Information

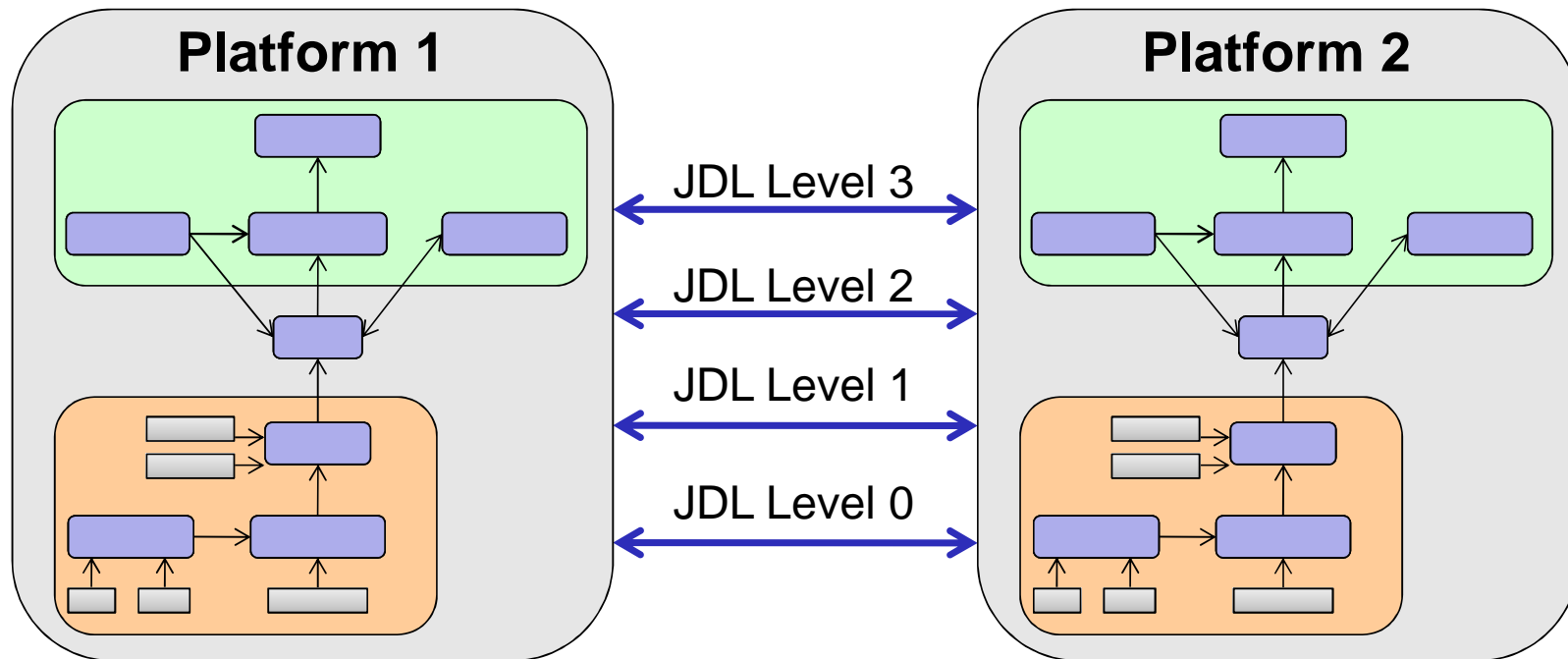


- Discover and broker information among agents.
- Refer to pieces of information among agents.
- Negotiate to make agents generate desired information.
- Share information among agents.

DyKnow Federations








Sharing and Fusing Information



It's a Streaming World!

It's a Streaming World! 1/2

- Oil operations 
- Traffic 
- Financial markets 
- Social networks 
- Generate data streams! 



<http://emanueledellavalle.org/Teaching/srt2015.html>

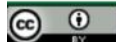
6

It's a Streaming World! 2/2

- What is the expected time to failure when that turbine's barring starts to vibrate as detected in the last 10 minutes?
- Is public transportation where the people are?
- Who are the best available agents to route all these unexpected contacts about the tariff plan launched yesterday?
- Who is driving the discussion about the top 10 emerging topics ?



E. Della Valle, S. Ceri, F. van Harmelen, D. Fensel It's a Streaming World! Reasoning upon Rapidly Changing Information. IEEE Intelligent Systems 24(6): 83-89 (2009)



<http://emanueledellavalle.org/Teaching/srt2015.html>

7

Requirements on Streaming Systems 1/8

A system able to answer those queries must be able to

- handle **massive datasets**

- A typical oil production platform is equipped with about **400.000 sensors**



- Telecom data is the most pervasive data source in urban are, in Milano there are **1.8 million mobile users**



- A global contact centre of a Telecom operator counts **500 millions of clients**



- Facebook alone has **1.1 billion** of active **users**



<http://emanueledellavalle.org/Teaching/srt2015.html>

8

Requirements on Streaming Systems 2/8

A system able to answer those queries must be able to

- process **data streams** on the fly
 - The sensors on typical oil production platform generates **10,000** observations per minute with **peaks of 100,000 o/m**
 - The mobile users in Milano generates **20,000** call/sms/data connections per minute with **peaks of 80,000 c/m**
 - A global contact centre receives **10,000** contacts per minute with **peaks of 30,000 c/m**
 - Facebook, as of May 2013, observes **3 millions "I like" per minute**



<http://emanueledellavalle.org/Teaching/srt2015.html>

9

Requirements on Streaming Systems 3/8

A system able to answer those queries must be able to

- cope with **heterogeneous dataset**

- The sensors on typical oil production have been deployed over 10 years by **10s of different producers**

- **Tens of data sources** are normally needed to make sense of an urban phenomena

- A global contact centre consists in **100s of offices** owned by different subsidiary companies **engaged yearly**

- Each social network has **its own data model, APIs, ...**



<http://emanueledellavalle.org/Teaching/srt2015.html>

10

Requirements on Streaming Systems 4/8

A system able to answer those queries must be able to

- cope with **incomplete data**

- 10s of **sensors** and networking links **broke down** daily

- **Coverage is incomplete**

- Only standard cases are covered by fully machine processable data records
100s of contacts per minute are manage ad-hoc

- **Conversations** happen **outside the social networks**, too!



<http://emanueledellavalle.org/Teaching/srt2015.html>

11

Requirements on Streaming Systems 5/8

A system able to answer those queries must be able to

- cope with **noisy data**
 - **Sensor out-of-operating range**
 - **Faulty sensors**
 - **Agents misunderstand, get tired, ...**
 - **Irony, sarcasm, ...**



<http://emanueledellavalle.org/Teaching/srt2015.html>

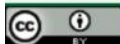
12

Requirements on Streaming Systems 6/8

A system able to answer those queries must be able to

- provide **reactive answers**

- detection of dangerous situations must occur within **minutes**
- recommendations to citizens must be performed in **few seconds**
- routing a contact through each step of the decision tree must take less than a **second**
- Search autocompleting may need to be updated every **few minutes**



<http://emanueledellavalle.org/Teaching/srt2015.html>

13

Requirements on Streaming Systems 7/8

A system able to answer those queries must be able to

- support **fine-grained information access**

- Identify **a turbine** among thousands



- Locate **a bus** among thousands



- Contact **an agent** among thousands



- Identify **an opinion maker** among thousands of influencers for a topic



<http://emanueledellavalle.org/Teaching/srt2015.html>

14

Requirements on Streaming Systems 8/8

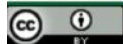
A system able to answer those queries must be able to

- integrate **complex domain models** of
 - **operational and control process**

- various **city aspects**

- **contact** management, **contract** types, **agent skills**, **contactor** profiles, ...

- **topics**, **user** profiles, ...



<http://emanueledellavalle.org/Teaching/srt2015.html>

15

40 ZETTABYTES

[43 TRILLION GIGABYTES]
of data will be created by 2020, an increase of 300 times from 2005



Volume SCALE OF DATA

It's estimated that **2.5 QUINTILLION BYTES** [2.3 TRILLION GIGABYTES] of data are created each day

Most companies in the U.S. have at least **100 TERABYTES** [100,000 GIGABYTES] of data stored



The FOUR V's of Big Data

From traffic patterns and music downloads to web history and medical records, data is recorded, stored, and analyzed to enable the technology and services that the world relies on every day. But what exactly is big data, and how can these massive amounts of data be used?

As a leader in the sector, IBM data scientists break big data into four dimensions: **Volume, Velocity, Variety and Veracity**

Depending on the industry and organization, big data encompasses information from multiple internal and external sources such as transactions, social media, enterprise content, sensors and mobile devices. Companies can leverage data to adapt their products and services to better meet customer needs, optimize operations and infrastructure, and find new sources of revenue.

By 2015
4.4 MILLION IT JOBS will be created globally to support big data, with 1.9 million in the United States



As of 2011, the global size of data in healthcare was estimated to be

150 EXABYTES [161 BILLION GIGABYTES]



30 BILLION PIECES OF CONTENT are shared on Facebook every month



Variety DIFFERENT FORMS OF DATA

By 2014, it's anticipated there will be **420 MILLION WEARABLE, WIRELESS HEALTH MONITORS**

4 BILLION+ HOURS OF VIDEO are watched on YouTube each month



400 MILLION TWEETS are sent per day by about 200 million monthly active users

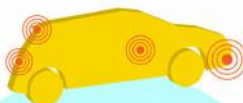


The New York Stock Exchange captures **1 TB OF TRADE INFORMATION** during each trading session



Velocity ANALYSIS OF STREAMING DATA

Modern cars have close to **100 SENSORS** that monitor items such as fuel level and tire pressure



By 2016, it is projected there will be **18.9 BILLION NETWORK CONNECTIONS** – almost 2.5 connections per person on earth



1 IN 3 BUSINESS LEADERS

don't trust the information they use to make decisions



Poor data quality costs the US economy around **\$3.1 TRILLION A YEAR**



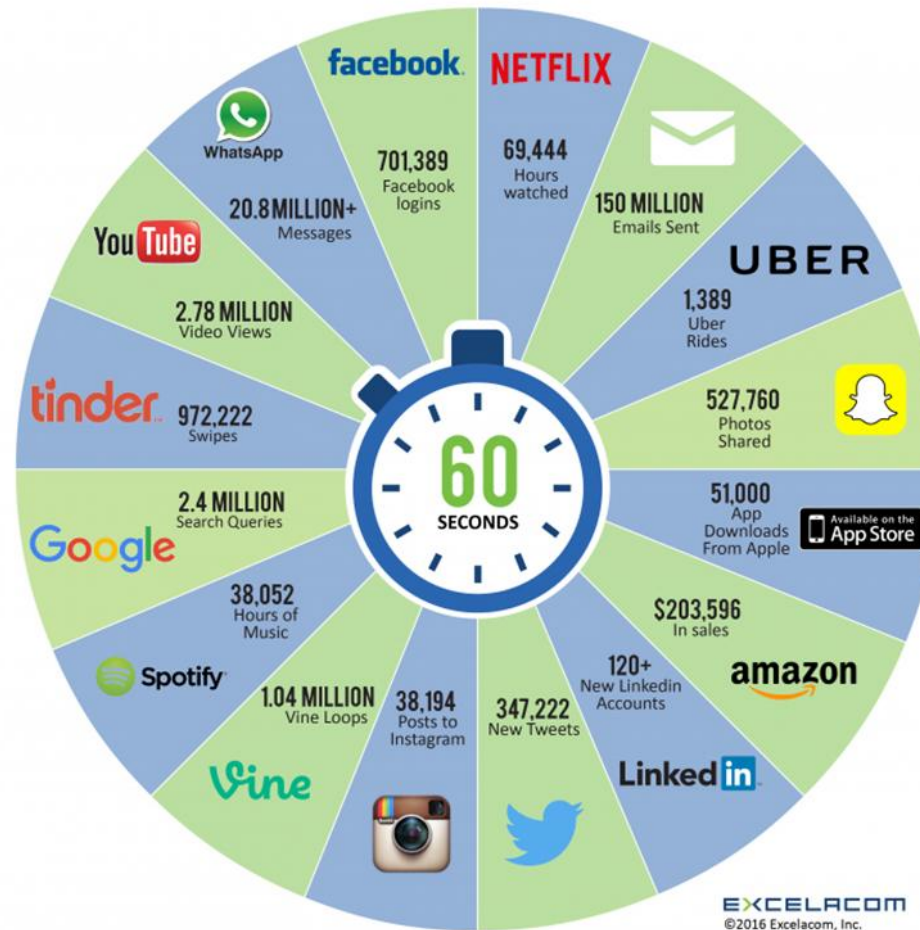
Veracity UNCERTAINTY OF DATA

27% OF RESPONDENTS

in one survey were unsure of how much of their data was inaccurate



2016 What happens in an INTERNET MINUTE?



Stream Reasoning Requirements Summary

A system able to answer those queries must be able to

- handle **massive datasets**
- process **data streams** on the fly
- cope with **heterogeneous dataset**
- cope with **incomplete data**
- cope with **noisy data**
- provide **reactive answers**
- support **fine-grained information access**
- integrate **complex domain models**

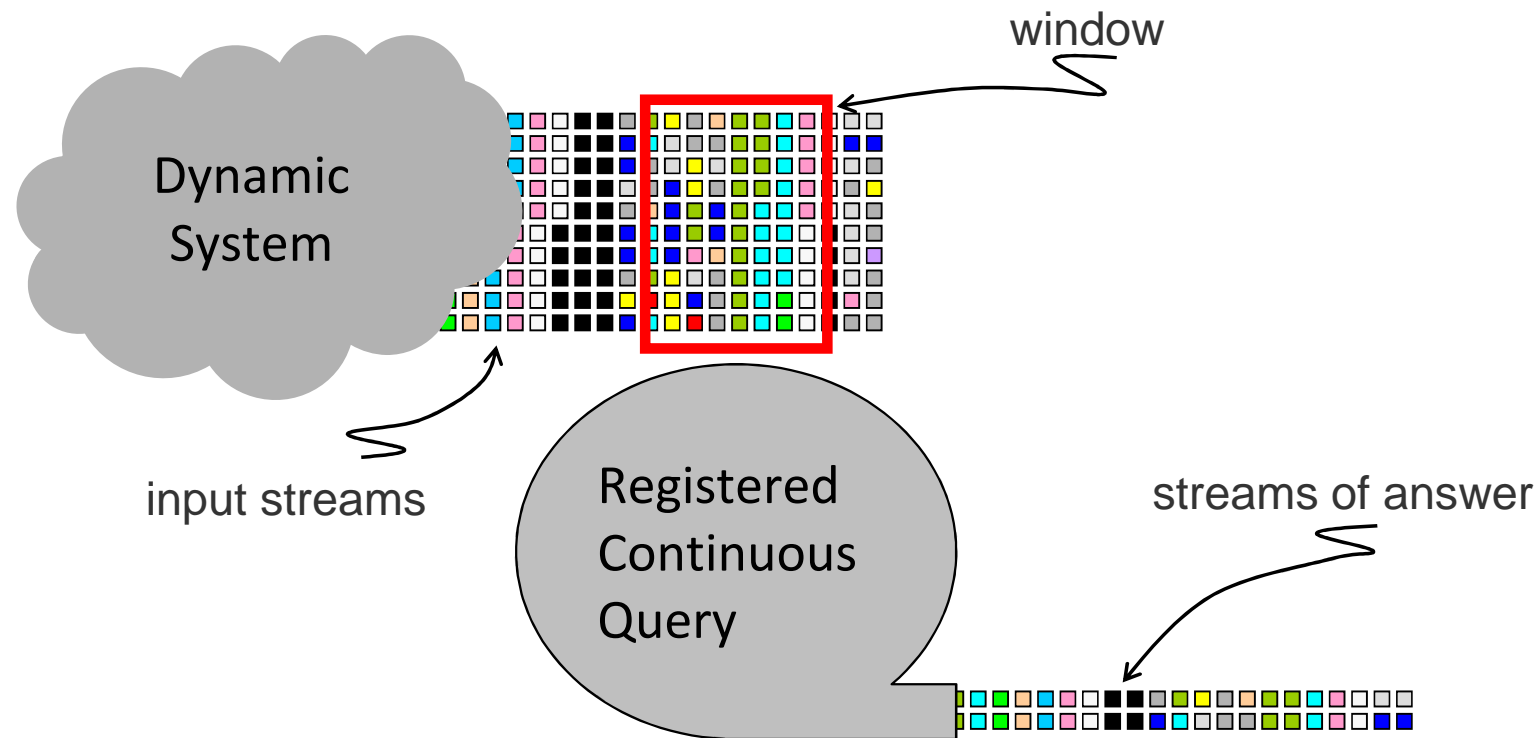
In **Big Data** terms →

X			
	X		
		X	
		X	X
			X
	X		
X	X		
		X	
Volume	Velocity	Variety	Veracity

Stream Reasoning Requires a Change of Paradigms!

- **From persistent data**
 - to be stored and queried on demand
 - a.k.a. one time semantics
- **To transient data**
 - to be consumed on the fly by continuous queries
 - a.k.a. continuous semantics

Continuous Semantics



Approaches to Stream Reasoning

Processing and Reasoning with Streams

- Traditional DBMSs
 - require data to be (persistently) stored and indexed before it could be processed and
 - process data only when explicitly asked by the users, that is, asynchronously with respect to its arrival.
- Stream Processing
 - Streams are usually unbounded.
 - No assumption can be made on data arrival order.
 - Size and time constraints make it difficult to store and process data stream elements after their arrival; one-time processing is the typical mechanism used to deal with streams.

Processing and Reasoning with Streams

- Two competing models
 - the *data stream processing* model [Babcock et al. 2002] and
 - the *complex event processing* model [Luckham 2001].
- DSMSs resemble DBMSs, especially in the way they process incoming data through a sequence of transformations based on common SQL operators, like selections, aggregates, joins, and all the operators defined in general by relational algebra.
- The complex event processing model views flowing information items as notifications of events happening in the external world, which have to be filtered and combined to understand what is happening in terms of higher-level events.
 - Accordingly, the focus of this model is on detecting occurrences of particular patterns of (low-level) events that represent the higher-level events whose occurrence has to be notified to the interested parties.

Stream Reasoning – Approaches

- Data stream management systems
 - Create new streams according to the following specification
- Complex event processing
 - Detect complex events based on specifications
- Stream reasoning in the semantic web
 - Continuous SPARQL queries over streaming RDF triples
- Temporal logic-based stream reasoning
 - Run-time verification / Path checking
 - Verifying that a stream satisfies a temporal logical formula
- Data flow systems
- Functional reactive programming

Stream Reasoning – Approaches

- **Window-based**, taking streams and turning them into relations over which standard reasoning/processing can be made. DSMS and RSP are both window-based.
- **Event-based**, define and detect complex events in streams of events. Recognizing complex events ought to be a form of reasoning, it infers implicit information, which is a reasonable definition of reasoning.
- **Logic-based**, evaluate temporal logical formulas over infinite streams.
- A Unified Model?
 - It seems that window-based and logic-based can be combined through window-based states.
 - It seems that window-based and event-based can be combined, see e.g. EP-SPARQL.

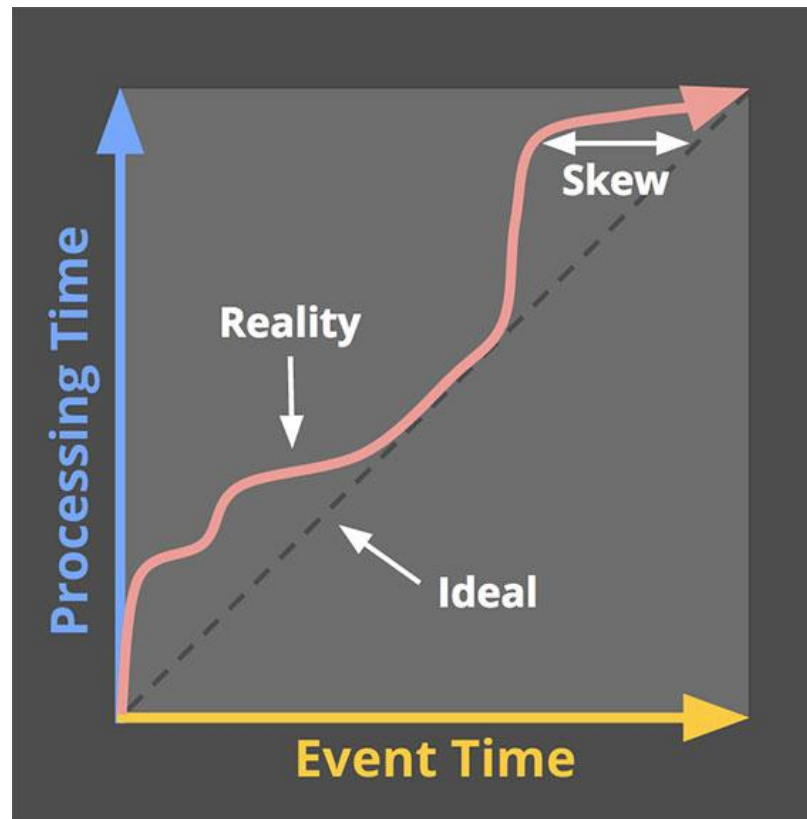
Representing Streams

- A stream is a potentially infinite sequence of elements.
- Timed vs Untimed
 - A potentially infinite sequence of elements $\langle e_1, e_2, e_3, \dots \rangle$, implicit total order, no explicit time
 - A potentially infinite sequence of time-stamped elements $\langle \langle t_1, e_1 \rangle, \dots \rangle$ or $\langle e_1, e_2, e_3, \dots \rangle$ where e_1 is a tuple containing at least a time-stamp and the sequence is ordered based on the time-stamp. If not, then can represent that tuples arrive in the “wrong” order.
- Typed vs Untyped
 - The elements could have a uniform type or have different types. In the first case, the streams are typed.
- Synchronous vs Asynchronous
- Single stream vs Multiple streams

Time

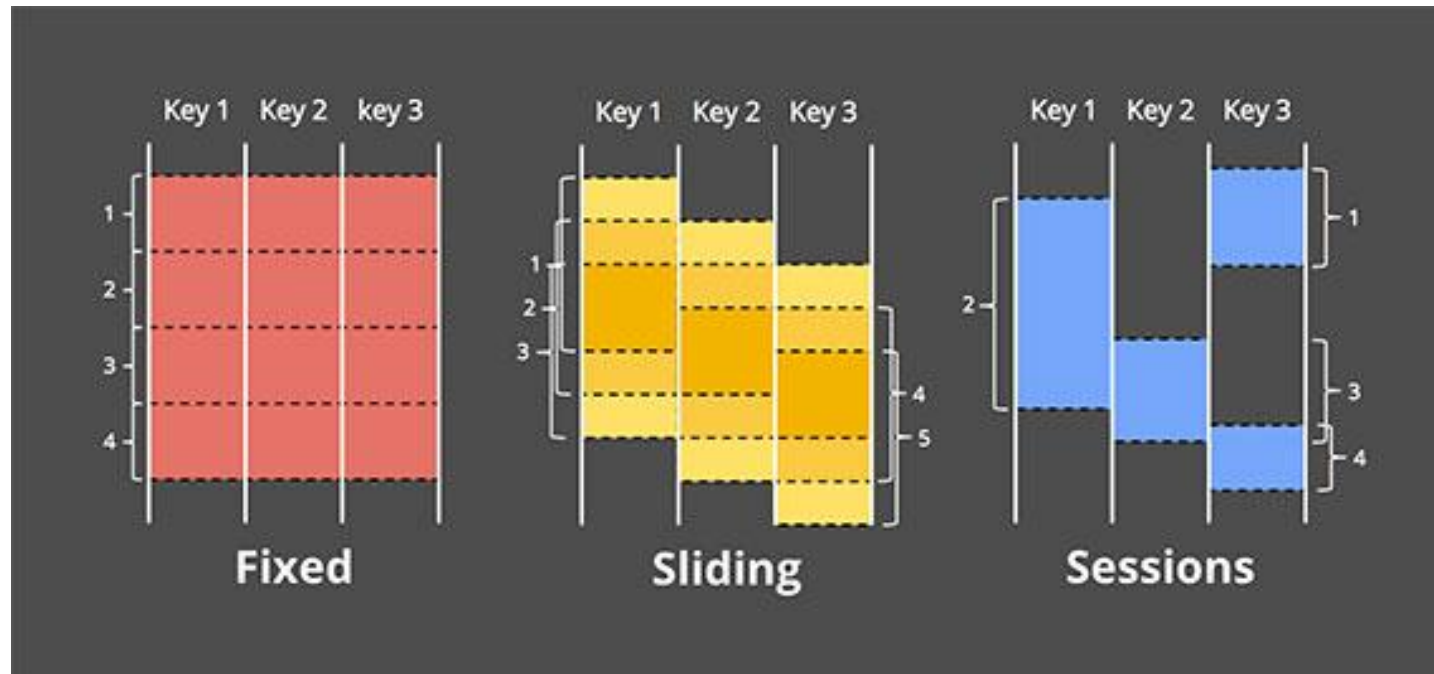
- Discrete time / Dense time
- Time-points / Time-intervals
- Relative time / Explicit time / Absolute time

Processing Time and Event Time

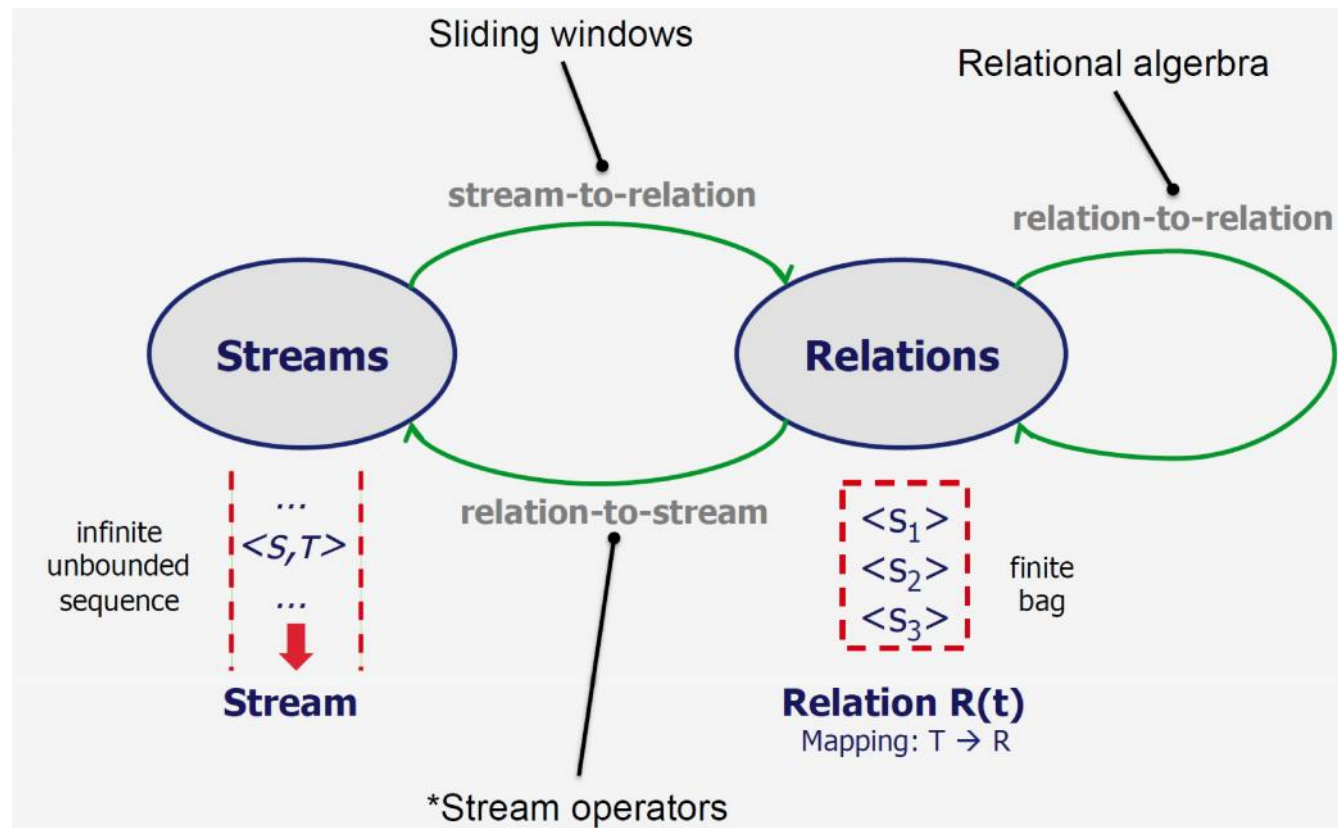


Windows

- Most stream based systems work on windows. They are basically sub-streams (and can be interpreted as locally closed worlds).



Stream Reasoning Through Windows



Logic-Based Approaches

- Linear time logics: $LTL < MTL < TPTL$
- Branching time logics: $CTL < CTL^*$
- Propositional vs Relational/First-order
- Discrete vs Dense time (point-wise and continuous)
 - Model checking: Decidable vs Undecidable
- Finite vs Infinite words (a stream is a finite prefix of an infinite word)

Model Checking vs Stream Reasoning

- Duality between specifying a system and Specifying the properties of the system
- Model checking: Timed automata model of a system + Formula \rightarrow True/False
- Path checking: Trace of the system + Formula \rightarrow True/False
- Stream reasoning: Stream + Formula \rightarrow True/False
- Stream reasoning as incremental formula evaluation
 - Compile to automata
 - Progression / Rewriting / Partial evaluation

Extensions / Research Challenges

- Asynchronous state information
- Implicit state information
 - Closed world assumption
- Atemporal static background information
- Incomplete state information
 - Missing state information
 - Disjunctive state information
- Uncertain state information (non-determinism)
- Incorrect state information
- Predicted / Anticipatory state information
- Time-varying domains/vocabulary
- Non-monotonicity

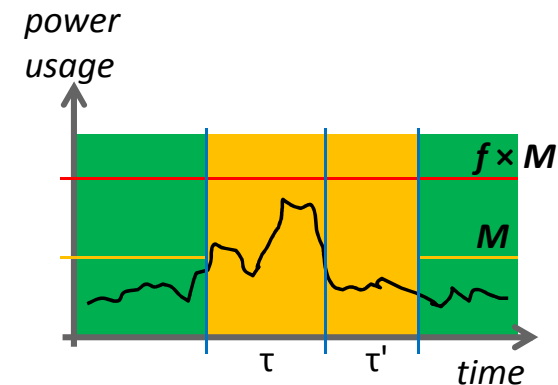
Spatio-Temporal Logic-Based Stream Reasoning

Metric Spatio-Temporal Logic (MSTL)

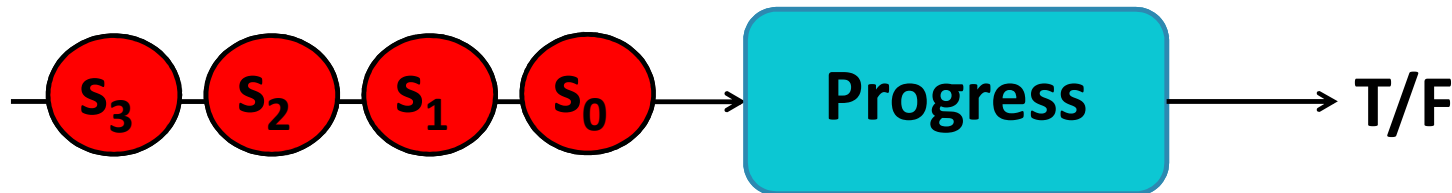
- Metric Spatio-Temporal Logic (**MSTL**) is a modal logic based on MTL and RCC-8 and uses the temporal operators
 - **F until_I G**
 - **always_I F** / **_I F** (equivalent to **F until_I false**)
 - **eventually_I F** / **$\diamond_I F$** (equivalent to true **until_I F**), and
 - **next** / **O**.
- Example: “It is always the case that if it rains, it will stop raining within 60 minutes”
 - **always** (rain \rightarrow **eventually** [0, 60min] (\neg rain))

Stream Reasoning using Metric Temporal Logic

always $\forall uav. ((\text{power_usage}(uav) > M) \rightarrow$
 $((\text{power_usage}(uav) < f \times M)$
 until $[0, \tau]$
 $(\text{always}[0, \tau'] \text{ power_usage}(uav) \leq M)))$



The semantics of these formulas is defined over infinite state sequences. Progression is one technique to check whether the current prefix is sufficient to determine the truth value of a formula.



Metric Temporal Logic

Definition 3 (Truth). *The statement that a spatio-temporal formula ϕ holds in $\mathcal{M} = \langle T, <, D, U, \mathcal{R}, V \rangle$ at time-point $t \in T$ is defined recursively:*

$\mathcal{M}, t \models f(o)$ iff $f(o) \in V(t)$ for any property $f(o)$

$\mathcal{M}, t \models \forall x \in s[f(x)]$ iff $\forall o \in \gamma(s) : \mathcal{M}, t \models f(o)$

$\mathcal{M}, t \models \exists x \in s[f(x)]$ iff $\exists o \in \gamma(s) : \mathcal{M}, t \models f(o)$

$\mathcal{M}, t \models \neg\phi$ iff $\mathcal{M}, t \not\models \phi$

$\mathcal{M}, t \models \phi \vee \psi$ iff $\mathcal{M}, t \models \phi$ or $\mathcal{M}, t \models \psi$

$\mathcal{M}, t \models \Box_{[t_1, t_2]} \phi$ iff $\forall t_1 \leq t' \leq t_2 : \mathcal{M}, t' \models \phi$

$\mathcal{M}, t \models \Diamond_{[t_1, t_2]} \phi$ iff $\exists t_1 \leq t' \leq t_2 : \mathcal{M}, t' \models \phi$

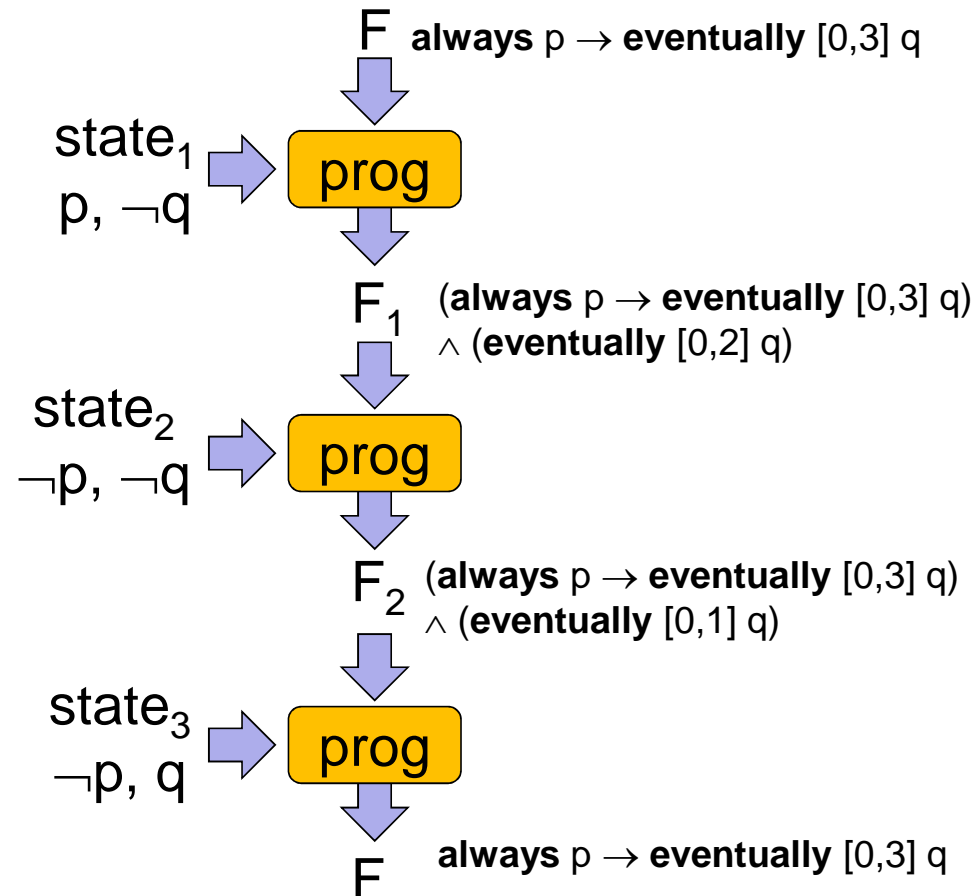
$\mathcal{M}, t \models \bigcirc\phi$ iff $\mathcal{M}, \text{succ}(t) \models \phi$

Progression of Metric Temporal Logic

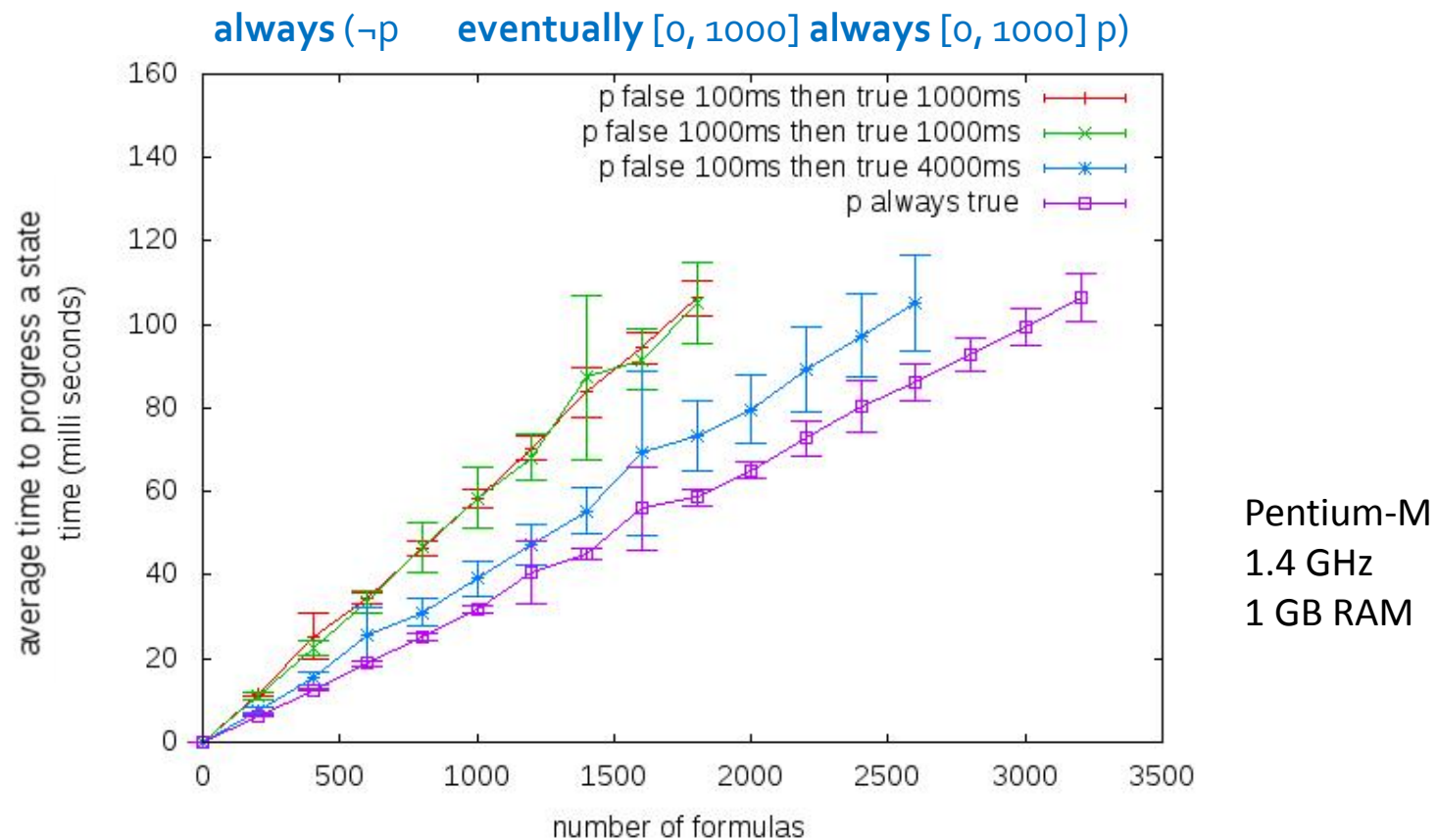
Definition 4 (*Progression of monitor formulas*) The following algorithm is used for progression of monitor formulas. Special cases for \Box and \Diamond can also be introduced for performance.

```
1 procedure Progress( $\phi, \tau, n, \mathcal{I}$ )
2 if  $\phi = f(\bar{x}) \hat{=} v$ 
3   if  $\mathcal{I} \models \text{Trans}([\tau] \phi)$  return  $\top$  else return  $\perp$ 
4 if  $\phi = \neg\phi_1$  return  $\neg\text{Progress}(\phi_1, \tau, n, \mathcal{I})$ 
5 if  $\phi = \phi_1 \otimes \phi_2$  return  $\text{Progress}(\phi_1, \tau, n, \mathcal{I}) \otimes \text{Progress}(\phi_2, \tau, n, \mathcal{I})$ 
6 if  $\phi = \forall x.\phi$  // where  $x$  belongs to the finite domain  $X$ 
7   return  $\bigwedge_{c \in X} \text{Progress}(\phi[x \mapsto c], \tau, n, \mathcal{I})$ 
8 if  $\phi = \exists x.\phi$  // where  $x$  belongs to the finite domain  $X$ 
9   return  $\bigvee_{c \in X} \text{Progress}(\phi[x \mapsto c], \tau, n, \mathcal{I})$ 
10 if  $\phi$  contains no tense operator
11   if  $\mathcal{I} \models \text{Trans}(\phi)$  return  $\top$  else return  $\perp$ 
12 if  $\phi = \phi_1 \cup_{[\tau_1, \tau_2]} \phi_2$ 
13   if  $\tau_2 < 0$  return  $\perp$ 
14   elseif  $0 \in [\tau_1, \tau_2]$  return  $\text{Progress}(\phi_2, \tau, n, \mathcal{I}) \vee$ 
15      $(\text{Progress}(\phi_1, \tau, n, \mathcal{I}) \wedge (\phi_1 \cup_{[\tau_1 - n, \tau_2 - n]} \phi_2))$ 
16   else return  $\text{Progress}(\phi_1, \tau, n, \mathcal{I}) \wedge (\phi_1 \cup_{[\tau_1 - n, \tau_2 - n]} \phi_2)$ 
```

Progression of Metric Temporal Logic

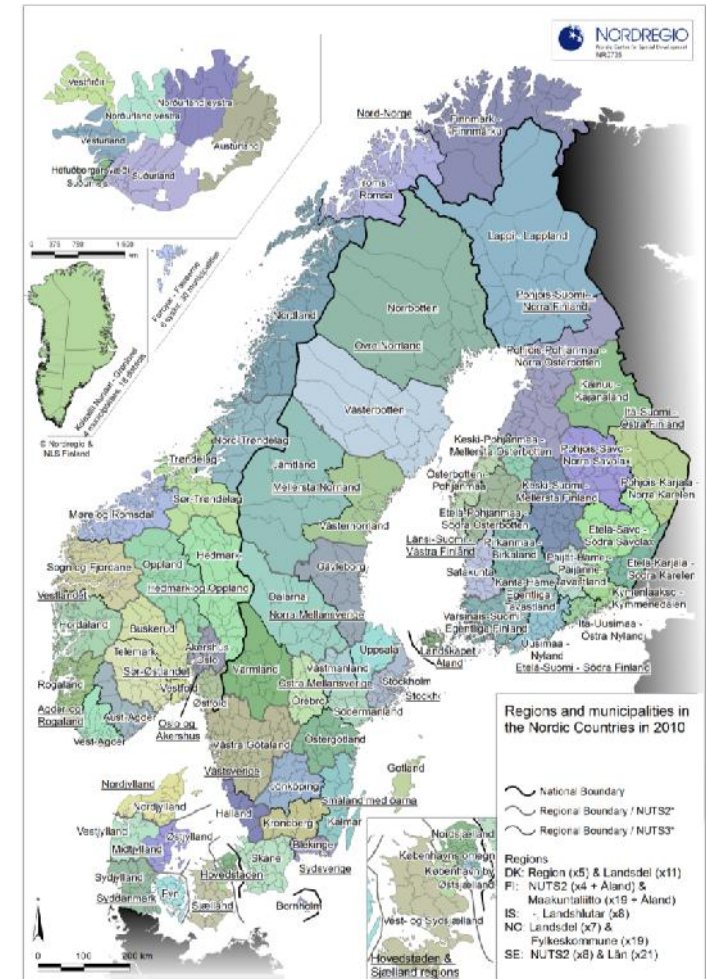


Stream Reasoning using Metric Temporal Logic



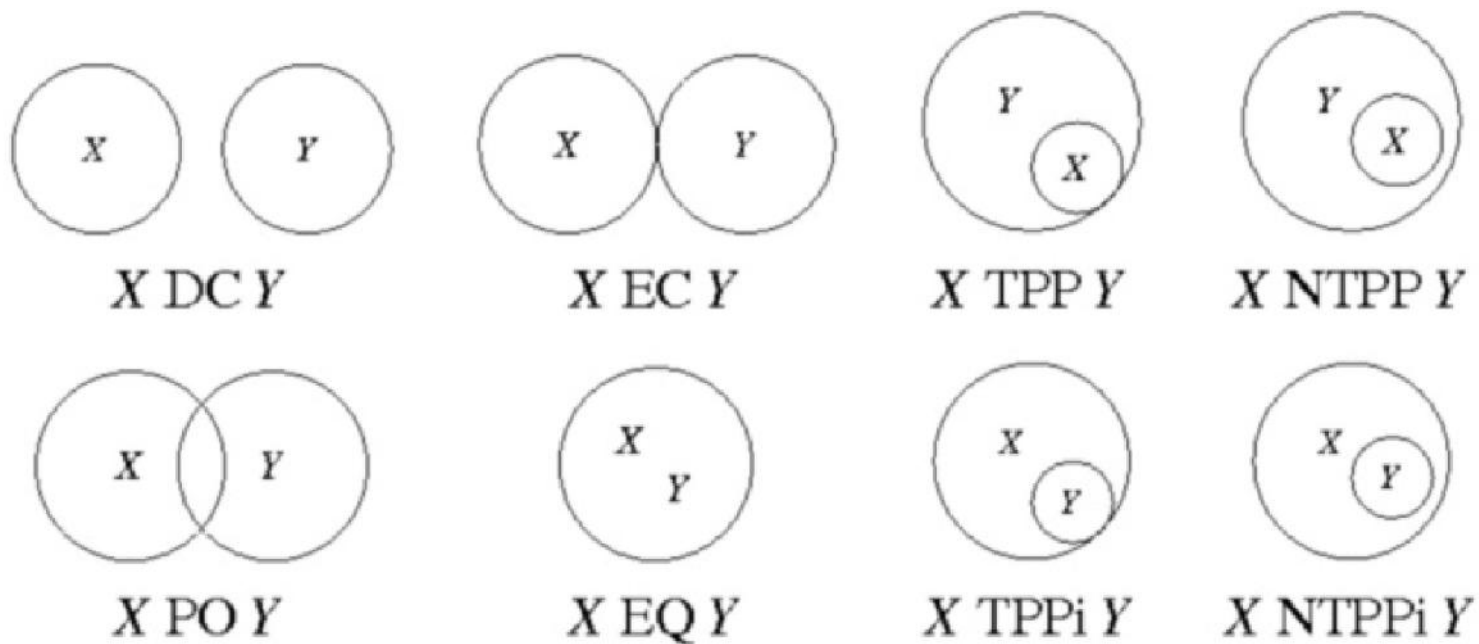
Qualitative Spatial Reasoning

- Qualitative spatial reasoning (QSR) deals with **regions** and relations between those regions called **spatial relations**.
- Qualitative spatial representation provides an abstract representation that handles imprecision.
- Natural to humans in term of communication.



Qualitative Spatial Reasoning with RCC-8

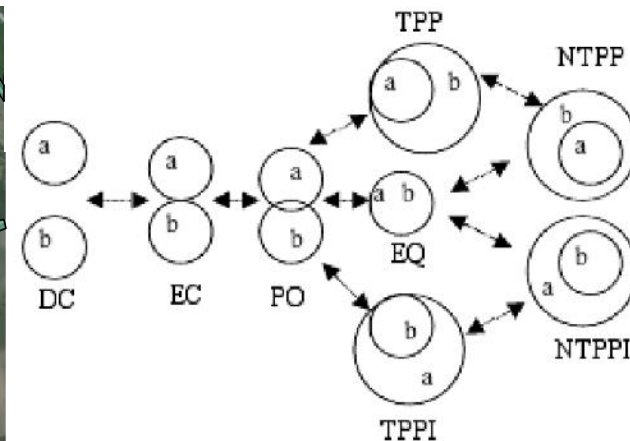
- The **Region Connection Calculus RCC-8** uses eight JEPD relations:



Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

- The temporal reasoning is extended with spatial reasoning using for example RCC-8. RCC-8 defines 8 primitive relations and a composition table for qualitative constraint reasoning based on path consistency.
- Allows expressing conditions such as:
 - $\forall uav, \text{restricted_area} \text{ always } DC(uav, \text{restricted_area})$
 - $\forall uav, \text{urban_area} \text{ always } (PO(uav, \text{urban_area}) \rightarrow \text{eventually } [0, 2\text{min}] \text{ altitude}(uav) > 100\text{m})$



Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

- The temporal reasoning is extended with spatial reasoning using for example RCC-8. RCC-8 defines 8 primitive relations and a composition table for qualitative constraint reasoning based on path consistency.
- Allows expressing conditions such as:
 - $\forall uav, restricted_area$ **always** $DC(uav, restricted_area)$
 - $\forall uav, urban_area$ **always** $(PO(uav, urban_area) \rightarrow \textbf{eventually} [0, 2min] altitude(uav) > 100m)$



DC(urban_area1, restricted_area1)
DC(urban_area1, restricted_area2)
DC(urban_area1, urban_area2)
DC(urban_area1, road1)
DC(urban_area1, road2)
EC(road1, restricted_area1)
EC(road1, restricted_area2)
PO(road1, urban_area2)
DC(road1, road2)
PO(uav1, road2)
PO(uav1, urban_area2) ...

Composition Table Region Connection Calculus RCC8

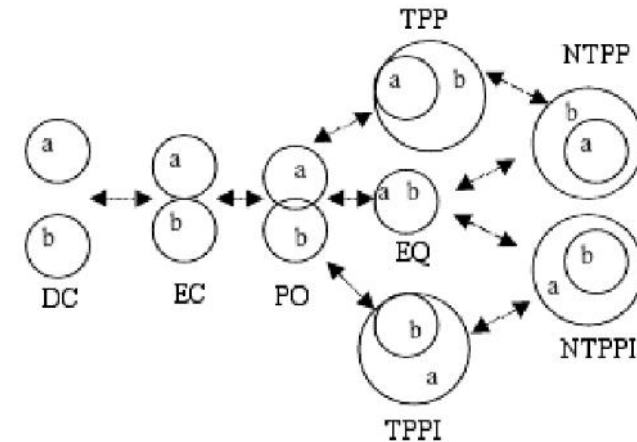
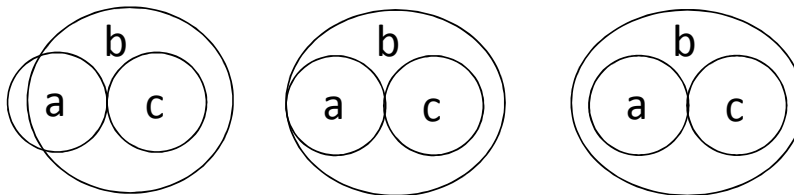
	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ
DC	*	DC,EC,PO,TPP,NTPP	DC,EC,PO,TPP,NTPP	DC,EC,PO,TPP,NTPP	DC,EC,PO,TPP,NTPP	DC	DC	DC
EC	DC,EC,PO,TPPi,NTPPi	DC,EC,PO,TPP,TPPi,EQ	DC,EC,PO,TPP,NTPP	EC,PO,TPP,NTPP	PO,TPP,NTPP	DC,EC	DC	EC
PO	DC,EC,PO,TPPi,NTPPi	DC,EC,PO,TPPi,NTPPi	*	PO,TPP,NTPP	PO,TPP,NTPP	DC,EC,PO,TPPi,NTPPi	DC,EC,PO,TPPi,NTPPi	PO
TPP	DC	DC,EC	DC,EC,PO,TPP,NTPP	TPP,NTPP	NTPP	DC,EC,PO,TPP,TPPi,EQ	DC,EC,PO,TPPi,NTPPi	TPP
NTPP	DC	DC	DC,EC,PO,TPP,NTPP	NTPP	NTPP	DC,EC,PO,TPP,NTPP	*	NTPP
TPPi	DC,EC,PO,TPPi,NTPPi	EC,PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO,TPP,TPPi,EQ	PO,TPP,NTPP	TPPi,NTPPi	NTPPi	TPPi
NTPPi	DC,EC,PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO,TPPi,NTPPi	PO,TPP,NTPP,TPPi,NTPPi,EQ	NTPPi	NTPPi	NTPPi
EQ	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ

Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

Known: $EC(a,c) \wedge NTPP(c,b)$

Deduced: $PO(a,b) \vee TPP(a,b)$
 $\vee NTPP(a,b)$



DC(urban_area1, restricted_area1)
 DC(urban_area1, restricted_area2)
 DC(urban_area1, urban_area2)
 DC(urban_area1, road1)
 DC(urban_area1, road2)
 EC(road1, restricted_area1)
 EC(road1, restricted_area2)
 PO(road1, urban_area2)
 DC(road1, road2)
 PO(uav1, road2)
 PO(uav1, urban_area2) ...

Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

- To handle incomplete spatial information we extend the first order logic to a three valued strong Kleene logic.

A and B	T	F	U	A or B	T	F	U	not A	
T	T	F	U	T	T	T	T	T	F
F	F	F	F	F	T	F	U	F	T
U	U	F	U	U	T	U	U	U	U

- The truth value of a spatial predicate $P(a,b)$ given a set S of disjunctive base relations that hold between a and b is:

- $P(a,b)$ is true if $P \in S$ and $|S|=1$
- $P(a,b)$ is unknown if $P \in S$ and $|S|>1$
- $P(a,b)$ is false if $P \notin S$
 $\text{always } (PO(a,b) \rightarrow \text{eventually } [0,2] DC(a,b))$

$\text{always } (PO(a,b) \rightarrow \text{eventually } [0,2] DC(a,b))$
 $\wedge (U \vee \text{eventually } [0,2] DC(a,b))$

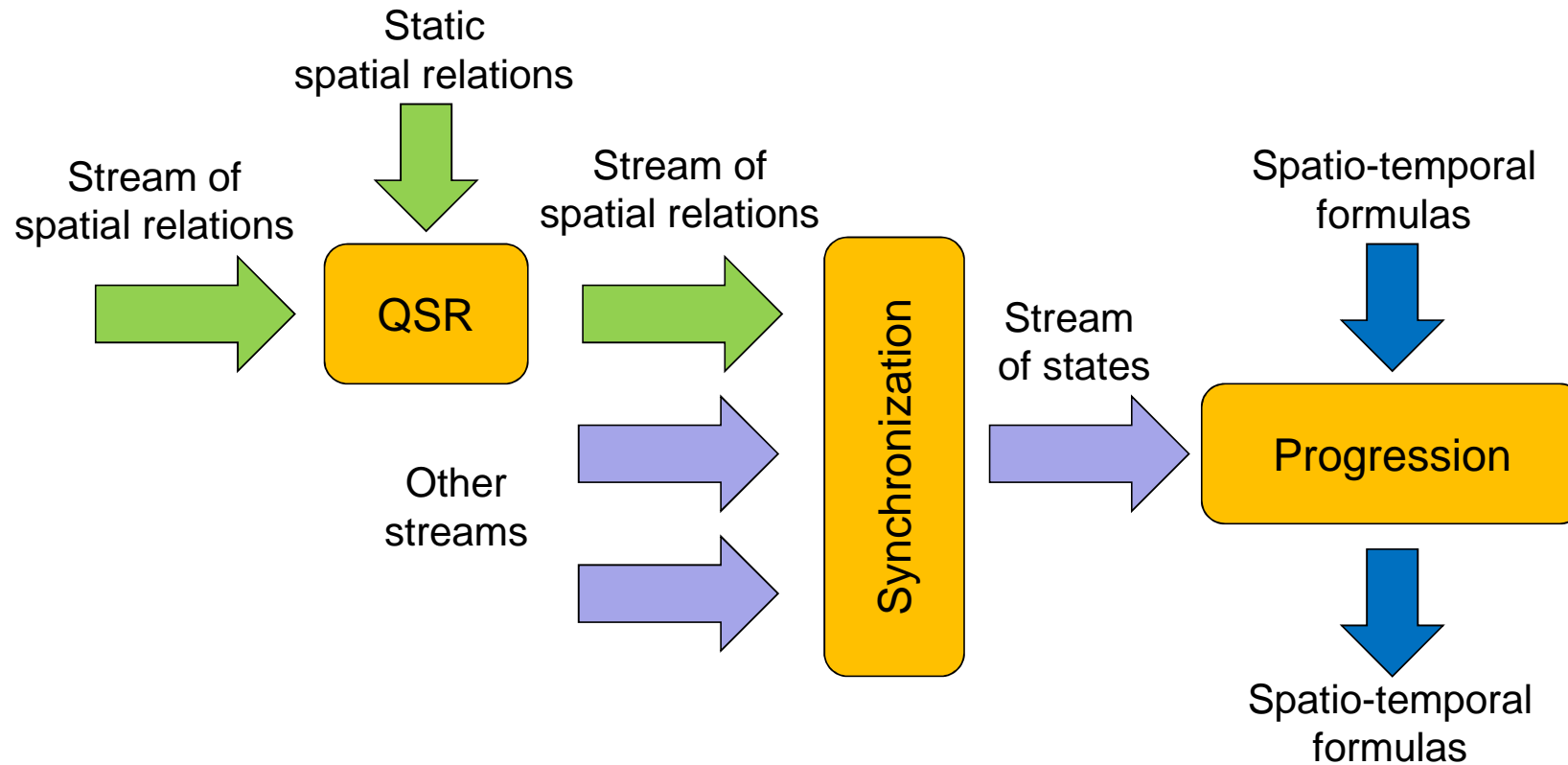
Known: $EC(a,c) \wedge NTPP(c,b)$

Deduced: $PO(a,b) \vee TPP(a,b)$
 $\vee NTPP(a,b)$

$PO(a,b) = U$ and $DC(a,b) = F$

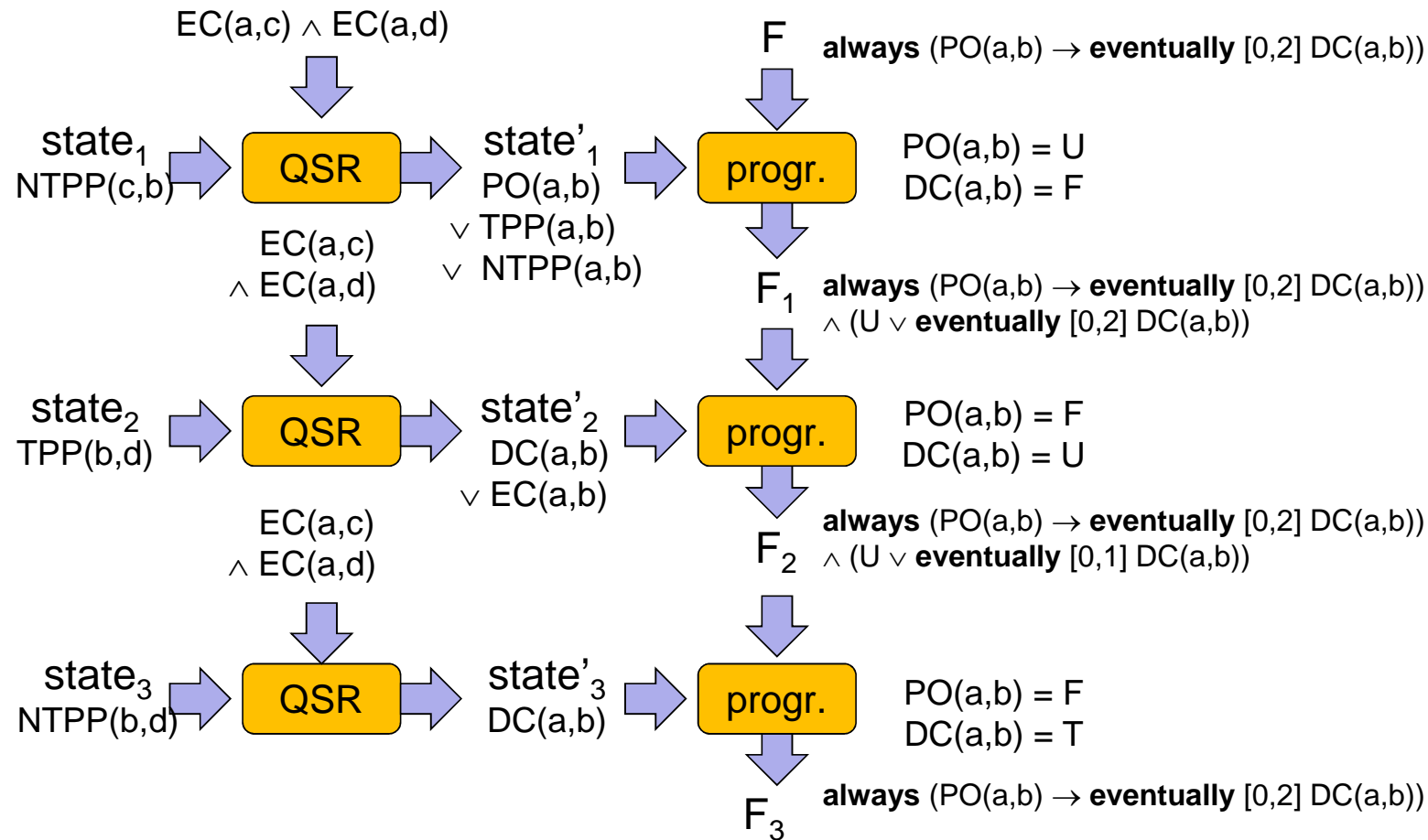
Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]



Progression of Metric Spatio-Temporal Logic

[Heintz and de Leng ECAI 2014]

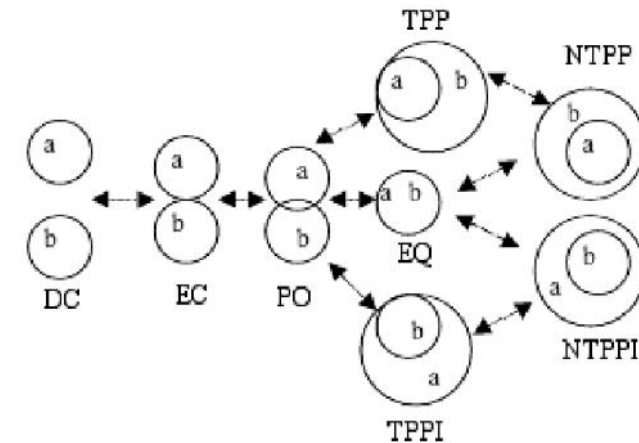
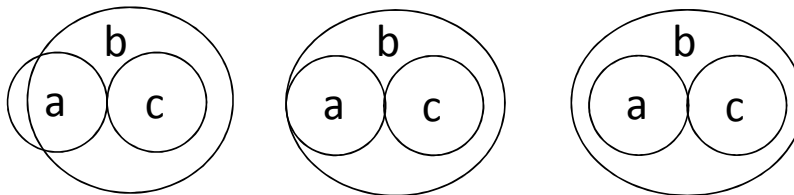


Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

Known: $EC(a,c) \wedge NTPP(c,b)$

Deduced: $PO(a,b) \vee TPP(a,b)$
 $\vee NTPP(a,b)$



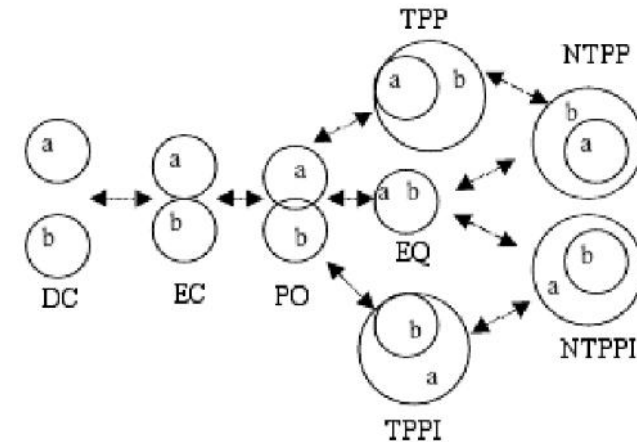
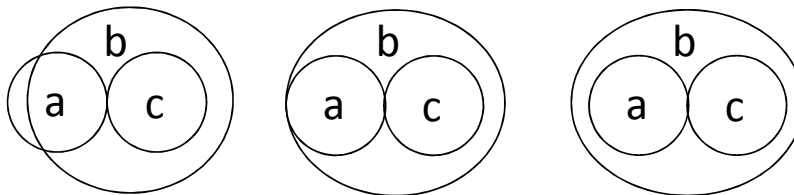
DC(urban_area1, restricted_area1)
 DC(urban_area1, restricted_area2)
 DC(urban_area1, urban_area2)
 DC(urban_area1, road1)
 DC(urban_area1, road2)
 EC(road1, restricted_area1)
 EC(road1, restricted_area2)
 PO(road1, urban_area2)
 DC(road1, road2)
 PO(uav1, road2)
 PO(uav1, urban_area2) ...

Spatio-Temporal Stream Reasoning in DyKnow

[Heintz and de Leng ECAI 2014]

Known: $EC(a,c) \wedge NTPP(c,b)$

Deduced: $PO(a,b) \vee TPP(a,b)$
 $\vee NTPP(a,b)$

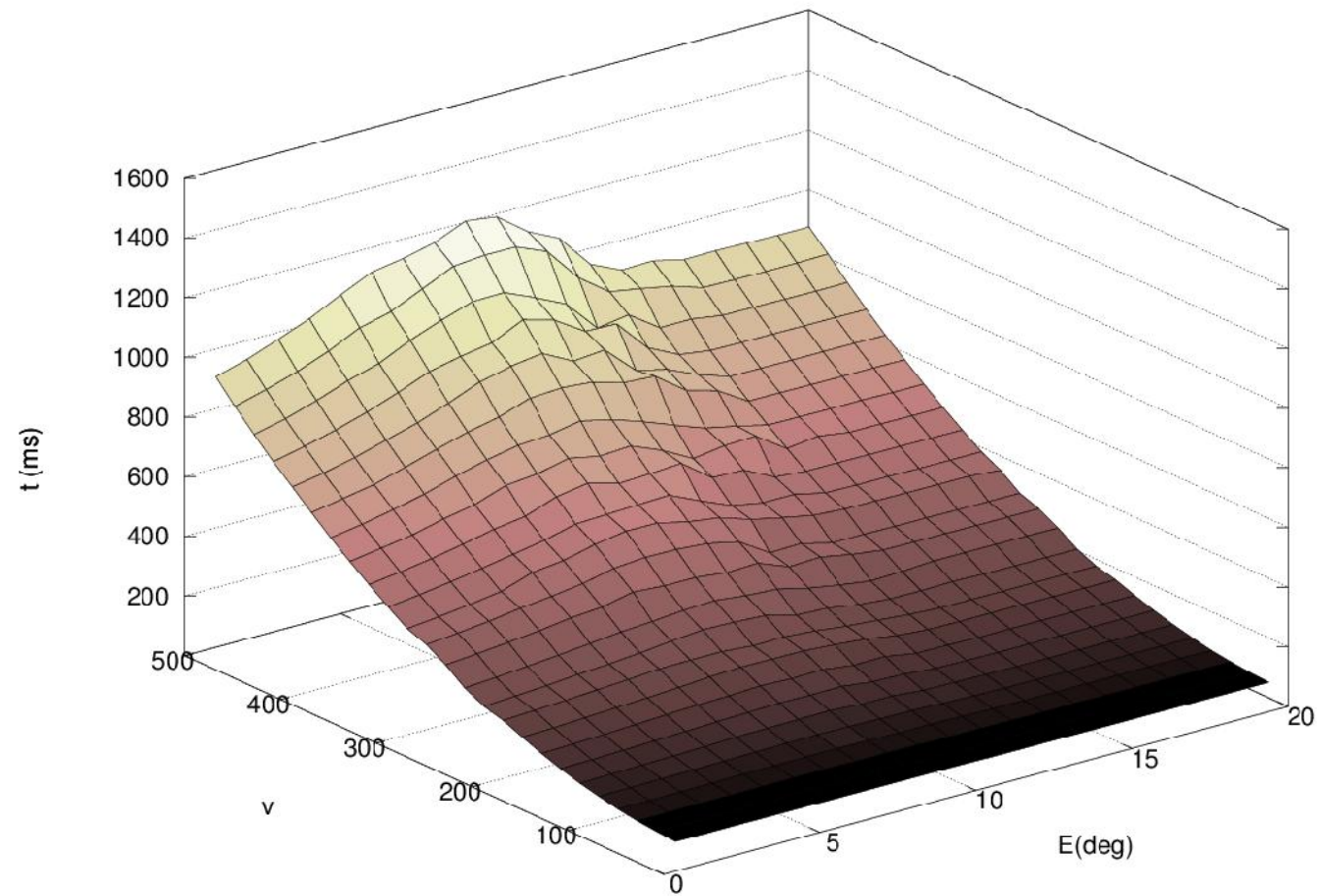


DC(urban_area1, restricted_area1)
 DC(urban_area1, restricted_area2)
 DC(urban_area1, urban_area2)
 DC(urban_area1, road1)
 DC(urban_area1, road2)
 EC(road1, restricted_area1)
 EC(road1, restricted_area2)
 PO(road1, urban_area2)
 DC(road1, road2)
 PO(uav1, road2)
 PO(uav1, urban_area2) ...

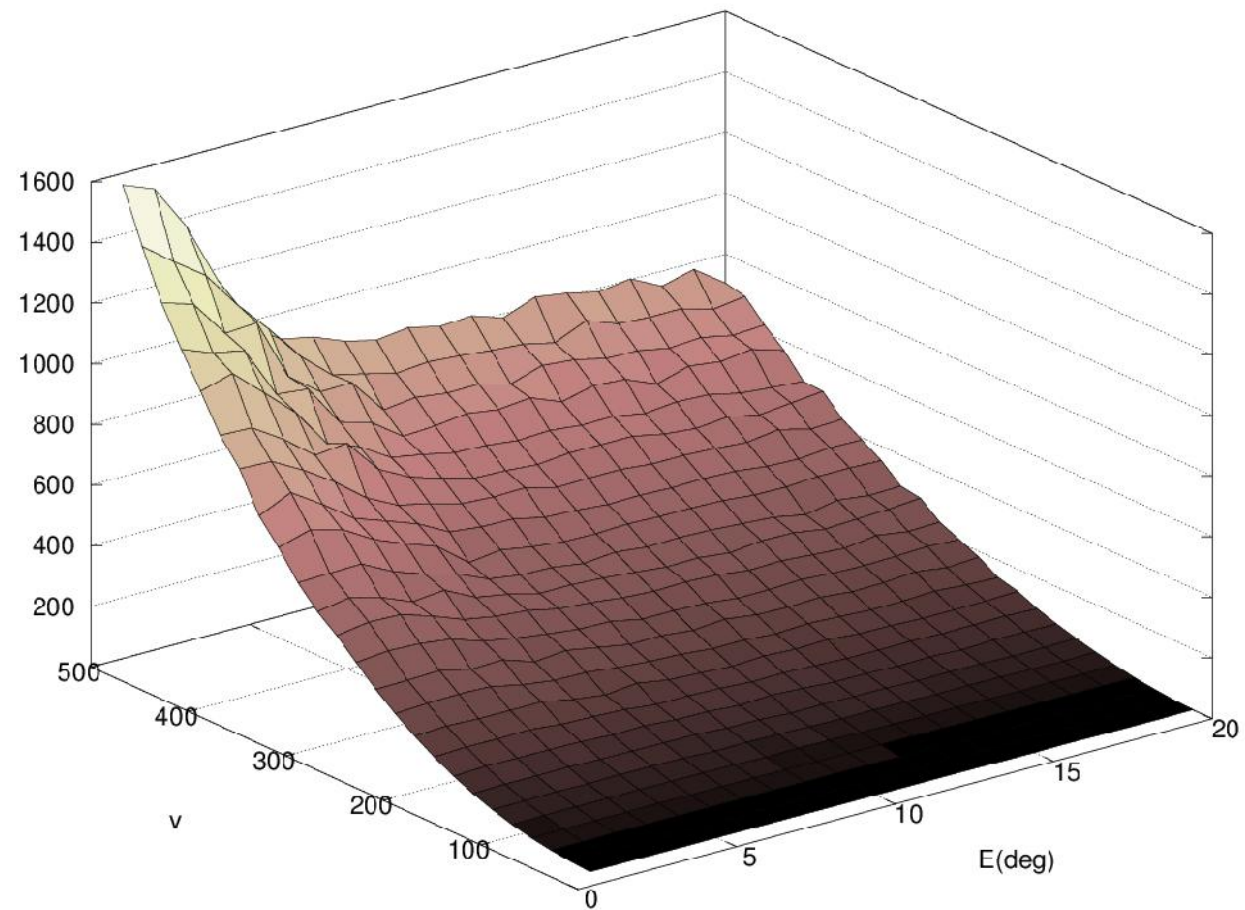
static

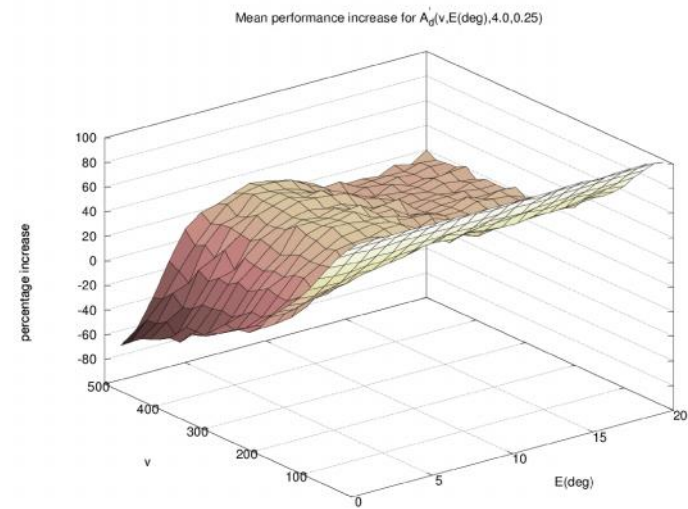
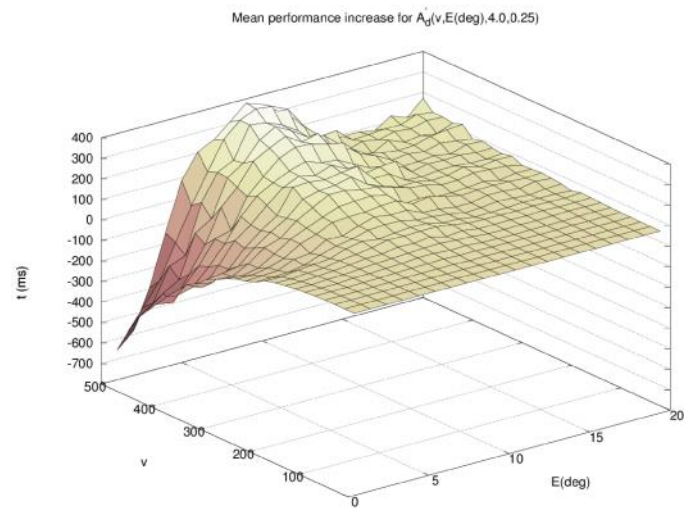
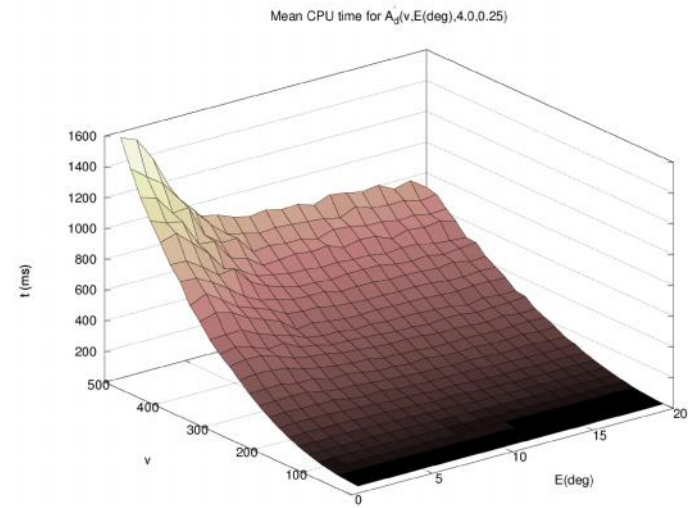
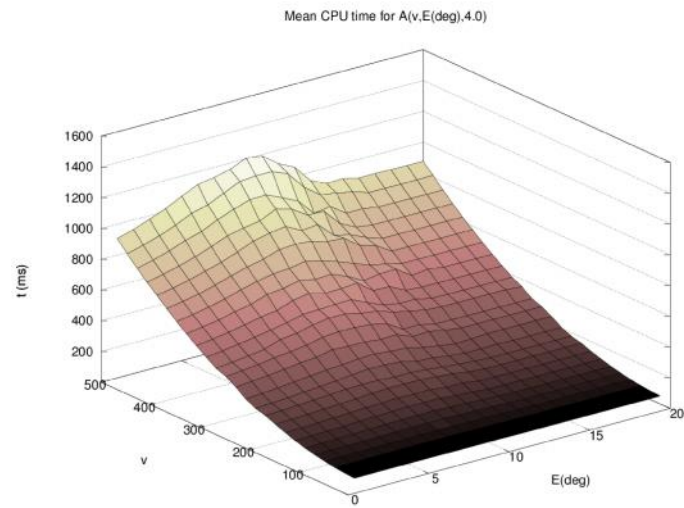
dynamic

Mean CPU time for $A(v, E(\text{deg}), 4.0)$



Mean CPU time for $A_d(v, E(\text{deg}), 4.0, 0.25)$





Spatio-Temporal Stream Reasoning in MSTL

[de Leng and Heintz AAI 2016]

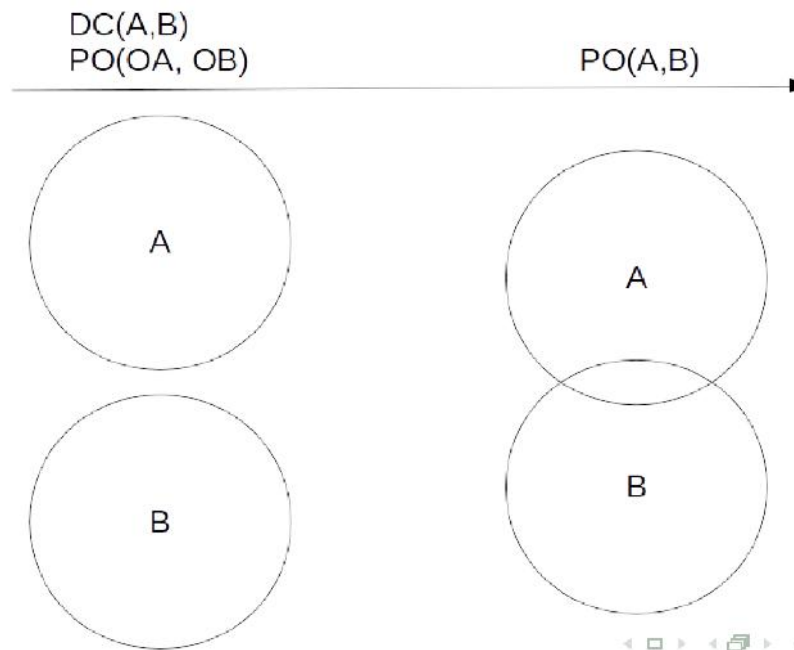
- We have recently extended MSTL to also consider regions as *spatial objects*.
- MSTL allows the **next** operator to be used over regions to denote the *region at the next time-point*.
- Examples: “It is always the case that if a car is speeding and tails another car, they will eventually collide”

$$\forall c_1 [\forall c_2 [c_1 \neq c_2 \wedge Car(c_1) \wedge Car(c_2) \rightarrow (\Box (PO(\bigcirc c_1, c_2) \wedge Speeding(c_1) \rightarrow \Diamond PO(c_1, c_2)))]]$$

Intertemporal Spatial Reasoning

[de Leng and Heintz AAAI 2016]

- Intertemporal spatial relations are problematic because they in many cases cannot be observed:

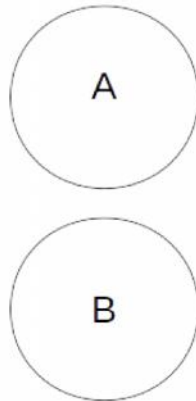


Intertemporal Spatial Reasoning

[de Leng and Heintz AAAI 2016]

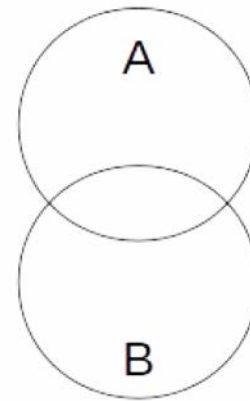
Observation at t_1 :

$$M^{t_1} = \begin{array}{c} A \quad B \\ \begin{bmatrix} \{EQ\} & \{DC\} \\ \{DC\} & \{EQ\} \end{bmatrix} \end{array}$$



Observation at t_2 :

$$M^{t_2} = \begin{array}{c} A \quad B \\ \begin{bmatrix} \{EQ\} & \{PO\} \\ \{PO\} & \{EQ\} \end{bmatrix} \end{array}$$



Intertemporal Spatial Reasoning

[de Leng and Heintz AAI 2016]

- Extended spatial relation matrices are used for describing intertemporal spatial relations:

$$M^{t_1 \cup t_2} = \begin{array}{c} \begin{array}{cc} & \begin{array}{cc} A & B & \bigcirc A & \bigcirc B \end{array} \\ \begin{array}{c} A \\ B \\ \bigcirc A \\ \bigcirc B \end{array} & \left[\begin{array}{cccc} \{EQ\} & \{DC\} & \mathcal{R}_8 & \mathcal{R}_8 \\ \{DC\} & \{EQ\} & \mathcal{R}_8 & \mathcal{R}_8 \\ \mathcal{R}_8 & \mathcal{R}_8 & \{EQ\} & \{PO\} \\ \mathcal{R}_8 & \mathcal{R}_8 & \{PO\} & \{EQ\} \end{array} \right] \end{array}$$

$$\mathcal{R}_8 = \{DC, PO, EC, EQ, TPP, TPPI, NTPP, NTPPI\}$$

Landmarks

[de Leng and Heintz AAI 2016]

Definition (Landmarks)

Landmark regions \mathcal{LM} are regions that are assumed to be **rigid**, i.e. $EQ(X, \bigcirc X)$. \mathcal{LM} is also called a **frame of reference**.



Landmarks

[de Leng and Heintz AAAI 2016]

Example for landmark $\mathcal{LM} = \{A\}$:

$$M^{t_1 \cup t_2} = \begin{bmatrix} \{EQ\} & \{DC\} & \{EQ\} & \mathcal{R}_8 \\ \{DC\} & \{EQ\} & \mathcal{R}_8 & \mathcal{R}_8 \\ \{EQ\} & \mathcal{R}_8 & \{EQ\} & \{PO\} \\ \mathcal{R}_8 & \mathcal{R}_8 & \{PO\} & \{EQ\} \end{bmatrix}$$

Landmarks

[de Leng and Heintz AAAI 2016]

Example for landmark $\mathcal{LM} = \{A\}$:

$$M^{t_1 \cup t_2} = \begin{bmatrix} \{EQ\} & \{DC\} & \{EQ\} & \{PO\} \\ \{DC\} & \{EQ\} & \{DC\} & \mathcal{R}_8 \setminus \{EQ\} \\ \{EQ\} & \{DC\} & \{EQ\} & \{PO\} \\ \{PO\} & \mathcal{R}_8 \setminus \{EQ\} & \{PO\} & \{EQ\} \end{bmatrix}$$

The Choice of Landmark Matters!

[de Leng and Heintz AAI 2016]

The choice of \mathcal{LM} affects the inferred intertemporal relations:

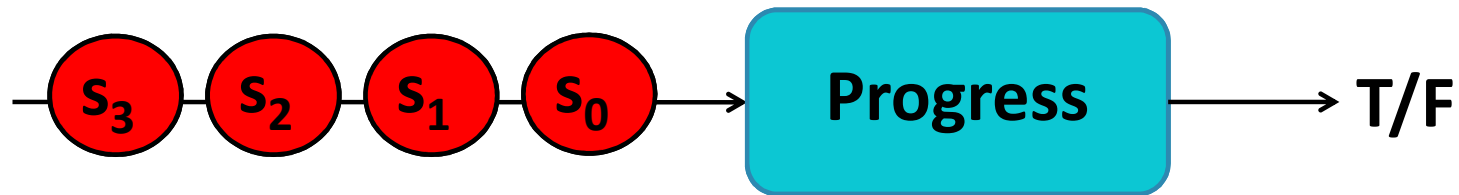
$$\mathcal{LM} = \{A\}, M^{t_1 \cup t_2} = \begin{bmatrix} \{EQ\} & \{DC\} & \{EQ\} & \{PO\} \\ \{DC\} & \{EQ\} & \{DC\} & \mathcal{R}_8 \setminus \{EQ\} \\ \{EQ\} & \{DC\} & \{EQ\} & \{PO\} \\ \{PO\} & \mathcal{R}_8 \setminus \{EQ\} & \{PO\} & \{EQ\} \end{bmatrix}$$

$$\mathcal{LM} = \{B\}, M^{t_1 \cup t_2} = \begin{bmatrix} \{EQ\} & \{DC\} & \mathcal{R}_8 \setminus \{EQ\} & \{DC\} \\ \{DC\} & \{EQ\} & \{PO\} & \{EQ\} \\ \mathcal{R}_8 \setminus \{EQ\} & \{PO\} & \{EQ\} & \{PO\} \\ \{DC\} & \{EQ\} & \{PO\} & \{EQ\} \end{bmatrix}$$

Progression of MSTL Formulas

[de Leng and Heintz AAI 2016]

- Progression does not look into the future...



- Rewriting rules for dealing with intertemporal relations:

$$R(\bigcirc X, \bigcirc Y) \Rightarrow \bigcirc R(X, Y)$$

$$R(X, \bigcirc Y) \Rightarrow \bigcirc R(\bigcirc^- X, Y)$$

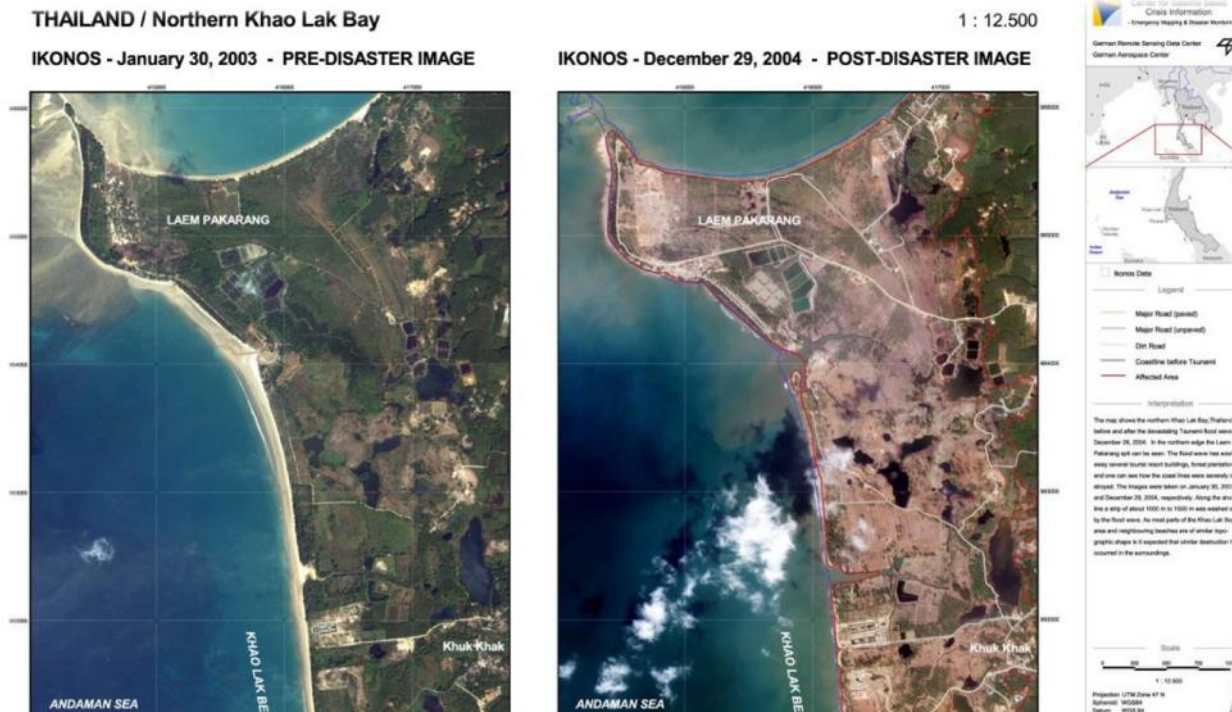
$$R(\bigcirc X, Y) \Rightarrow \bigcirc R(X, \bigcirc^- Y)$$

Summary

- We have presented the Metric Spatio-Temporal Logic MSTL which combines MTL with RCC-8.
- MSTL formulas can be incrementally evaluated over streams of states through progression.
- MSTL supports spatial reasoning both within and between time-points.
- MSTL supports incomplete information through a three-valued logic approach.
- The logic-based approach is very suitable for making safe autonomous systems through for example execution monitoring with formal guarantees.

Stream Reasoning for Safe Autonomous Systems

Application: Emergency Services Assistance



A devastating earthquake of high magnitude occurred on December, 26, 2004 off the west coast of Sumatra, Indonesia. The resulting Tsunami killed thousands of people in southern India, Sri Lanka, Indonesia, Thailand, etc.

Search and Relief

Searching for injured people and delivering food, medicine and other supplies are highly prioritized activities in disaster relief.



Finding Injured People



Finding Injured People - Mission



Assumptions

- Optimal flight altitude: 35- 50 m
- Average flight velocity: 5m/s
- Human body size: 20 - 50 pixels



Emergency Services Training School
Revinge, Sweden

1 Sq Km

Context

- 11 live bodies / 2 dummies
- 2 UAVs for scanning
- 290 x 185 m

EXAMPLE MISSION

Linköping University is a research-intensive university with a focus on innovation and entrepreneurship. The university's mission is to advance knowledge and create a sustainable future for society. This is achieved through research, education, and collaboration with the business sector. The university's research is characterized by a strong emphasis on innovation and entrepreneurship, and its education is designed to equip students with the skills and knowledge needed to succeed in a rapidly changing world.

ALGORITHM RESULTS

Finding Injured People - Result



Thermal and color images

Saliency Map

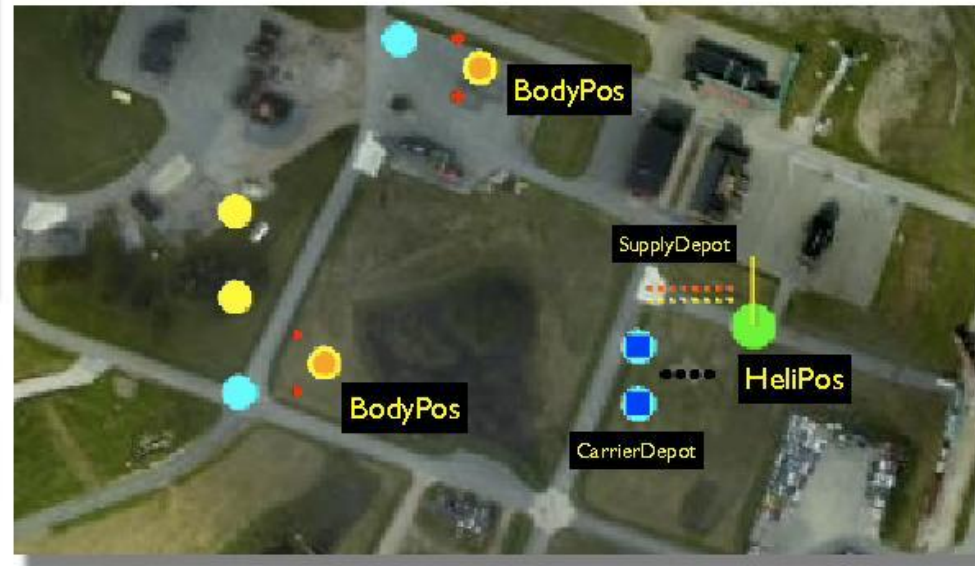


*The algorithm runs in
25 frames / second*

Deliver Food and Supplies



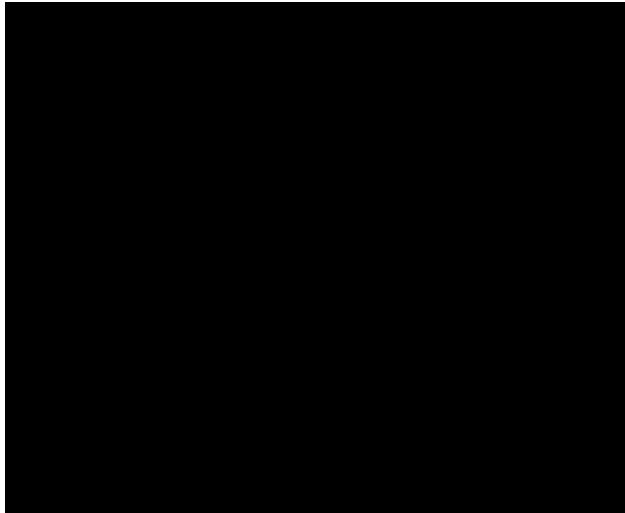
Search and Rescue Mission



Multiple UAVs, different carrying capacities

Multiple carrier and supply depots

Execution Monitoring



If things can go wrong they probably will!

This implies the need for continual monitoring of an autonomous system and its environment in a principled, contextual, task specific manner which can be specified by the system itself!

always (eventually [0, t] (always [0,t'] speed(uav) < T))

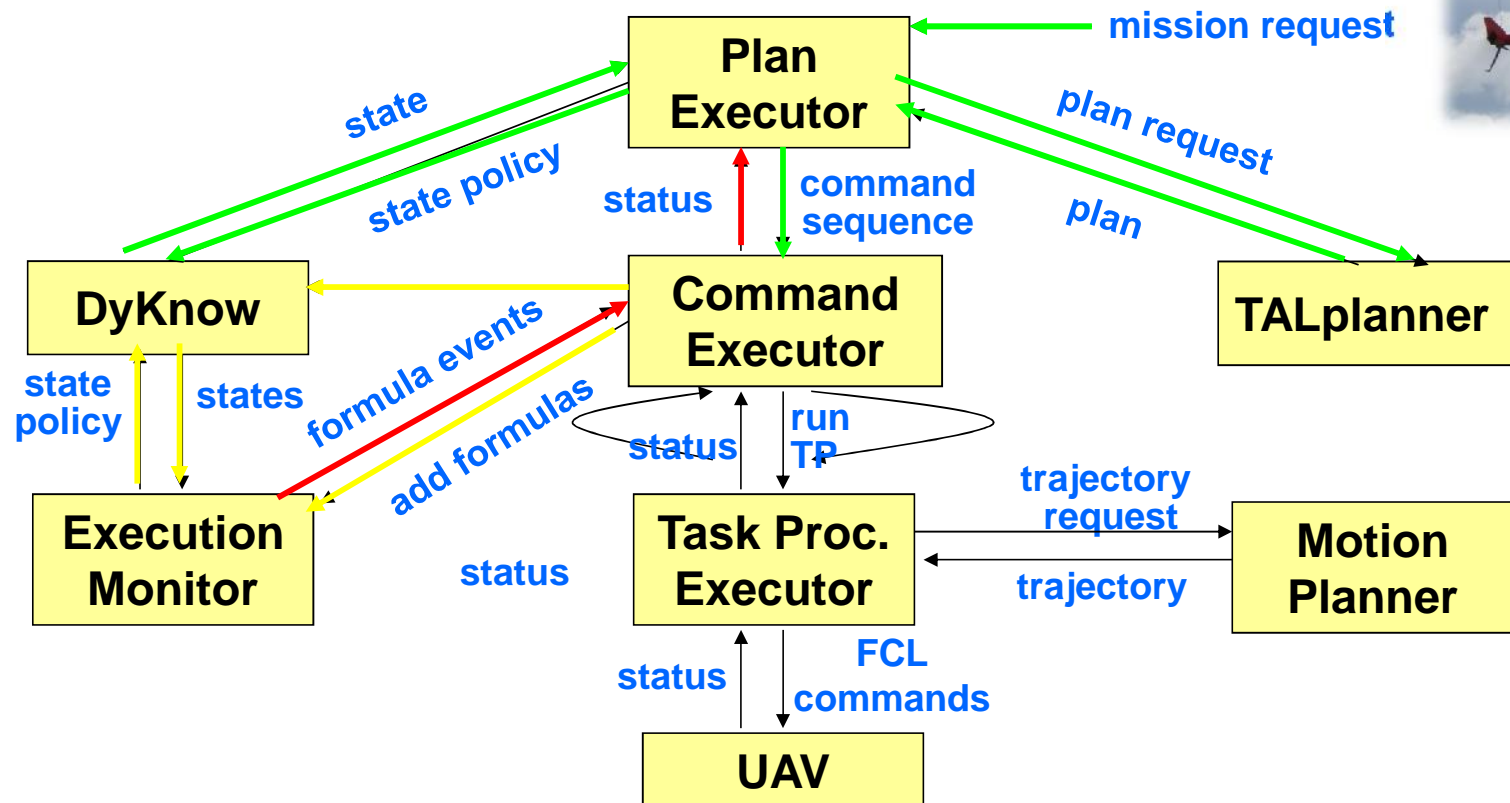
It should always be the case that within t time units from now, an interval of length t' should start where the UAV's speed stays below threshold T.

EXEC until [0, 5000] (\neg EXEC \wedge altitude(uav) > 7)

The command should take less than 5 seconds to execute and when the execution is finished the altitude of the UAV should be above 7 meters.

Planning, Plan Execution & Execution Monitoring

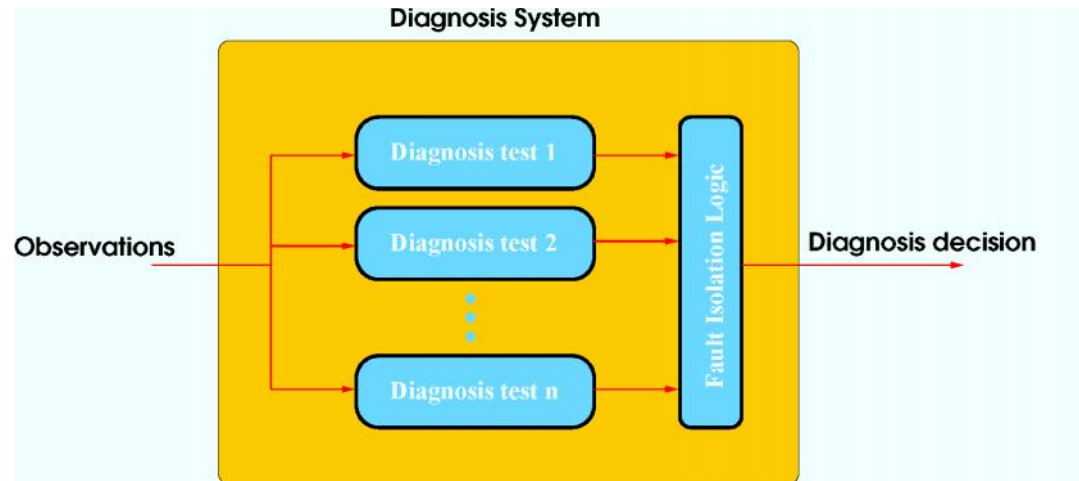
[Kvarnström, Heintz and Doherty ICAPS 2008; Kvarnström, Heintz and Doherty JAAMAS 2009]





DyKnow Application: **Diagnosis**

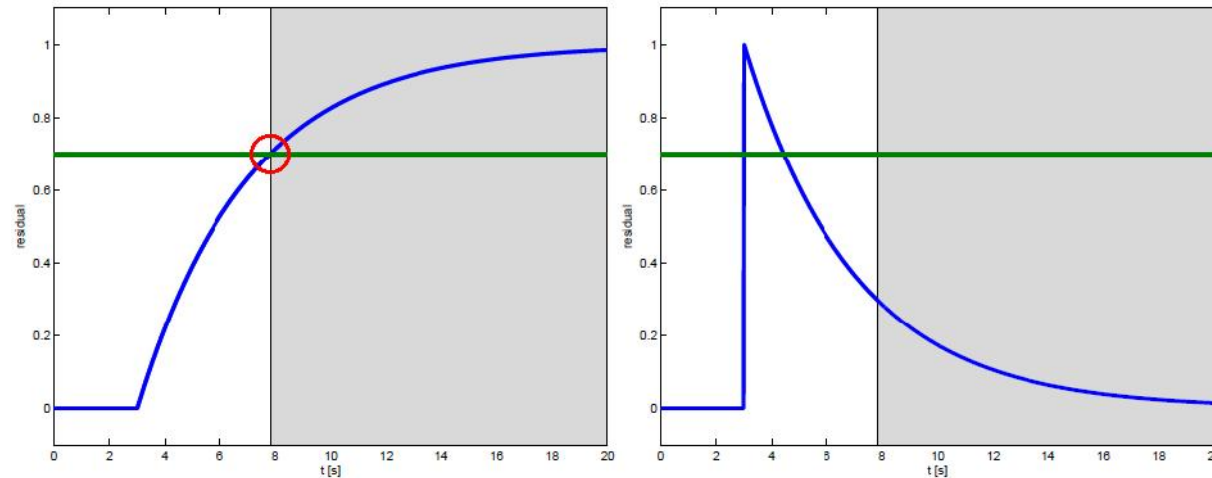
[Heintz, Krysander, Roll and Frisk DX 2008; Krysander, Heintz, Roll and Frisk CDC 2008 and EAAI 2010]



- **Problem:** High capability of distinguishing faults requires large number of tests and thus high on-line computational demands.
- **FlexDx**
 - Recognizes that not all test are needed at all times, e.g. no-fault case. Runs only tests needed at the moment.
 - The same capability of distinguishing faults as using all tests.

DyKnow Application: **Diagnosis**

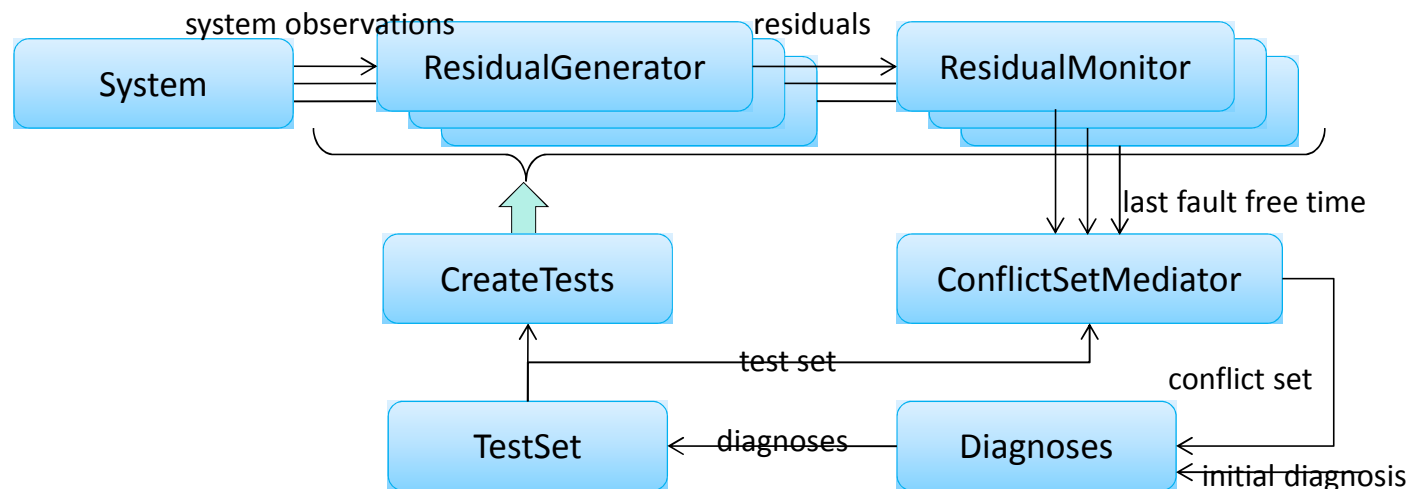
[Heintz, Krysander, Roll and Frisk DX 2008; Krysander, Heintz, Roll and Frisk CDC 2008 and EAAI 2010]



- **Problem:** High capability of distinguishing faults requires large number of tests and thus high on-line computational demands.
- **FlexDx**
 - Recognizes that not all test are needed at all times, e.g. no-fault case. Runs only tests needed at the moment.
 - The same capability of distinguishing faults as using all tests.

DyKnow Application: **Diagnosis**

[Heintz, Krysander, Roll and Frisk DX 2008; Krysander, Heintz, Roll and Frisk CDC 2008 and EAAI 2010]



- **Problem:** High capability of distinguishing faults requires large number of tests and thus high on-line computational demands.
- **FlexDx**
 - Recognizes that not all test are needed at all times, e.g. no-fault case. Runs only tests needed at the moment.
 - The same capability of distinguishing faults as using all tests.

DyKnow Application: **Diagnosis**

[Heintz, Krysander, Roll and Frisk DX 2008; Krysander, Heintz, Roll and Frisk CDC 2008 and EAAI 2010]

- 4 states ($\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2$)
- 1 actuator $u(t)$ and 3 sensors $y_i(t)$
- 6 single faults, one for each model equation
- measurement noise, $\nu_i(t)$
- 13 minimal tests (corresponds to submodels with redundancy 1)

$$J_1 \ddot{\theta}_1(t) = ku(t) - \alpha_1 \dot{\theta}_1(t) - M_s(t)$$

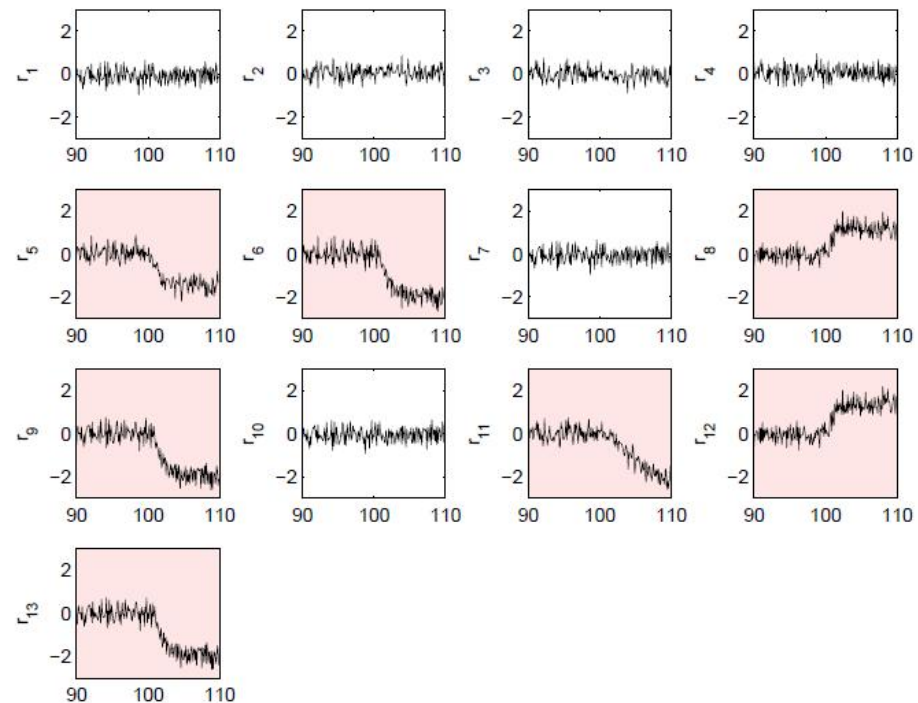
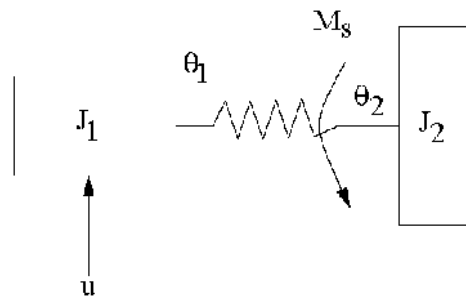
$$M_s(t) = \alpha_2(\theta_1(t) - \theta_2(t)) + \alpha_3(\dot{\theta}_1(t) - \dot{\theta}_2(t))$$

$$J_2 \ddot{\theta}_2(t) = -\alpha_4 \dot{\theta}_2(t) + M_s(t)$$

$$y_1 = \theta_1 + \nu_1(t)$$

$$y_2 = \dot{\theta}_1 + \nu_2(t)$$

$$y_3 = \theta_2 + \nu_3(t)$$



Fault occurs at time $t=100$, plot $t \in [90, 110]$.

DyKnow Application: **Diagnosis**

[Heintz, Krysander, Roll and Frisk DX 2008; Krysander, Heintz, Roll and Frisk CDC 2008 and EAAI 2010]

- 4 states ($\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2$)
- 1 actuator $u(t)$ and 3 sensors $y_i(t)$
- 6 single faults, one for each model equation
- measurement noise, $\nu_i(t)$
- 13 minimal tests (corresponds to submodels with redundancy 1)

$$J_1 \ddot{\theta}_1(t) = ku(t) - \alpha_1 \dot{\theta}_1(t) - M_s(t)$$

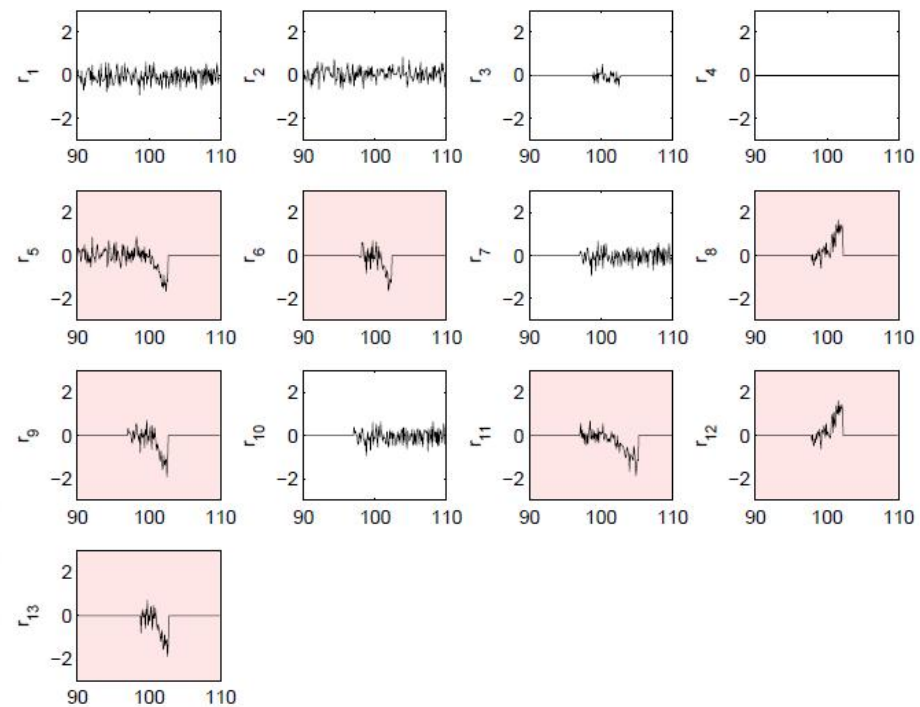
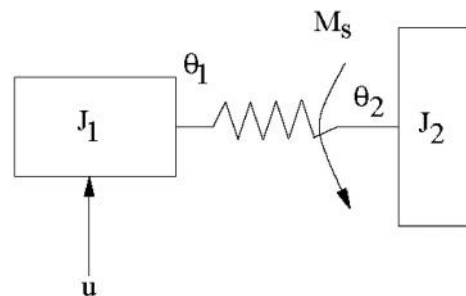
$$M_s(t) = \alpha_2(\theta_1(t) - \theta_2(t)) + \alpha_3(\dot{\theta}_1(t) - \dot{\theta}_2(t))$$

$$J_2 \ddot{\theta}_2(t) = -\alpha_4 \dot{\theta}_2(t) + M_s(t)$$

$$y_1 = \theta_1 + \nu_1(t)$$

$$y_2 = \dot{\theta}_1 + \nu_2(t)$$

$$y_3 = \theta_2 + \nu_3(t)$$



Fault occurs at time $t=100$, plot $t \in [90, 110]$.

Grounding Logic-Based Stream Reasoning

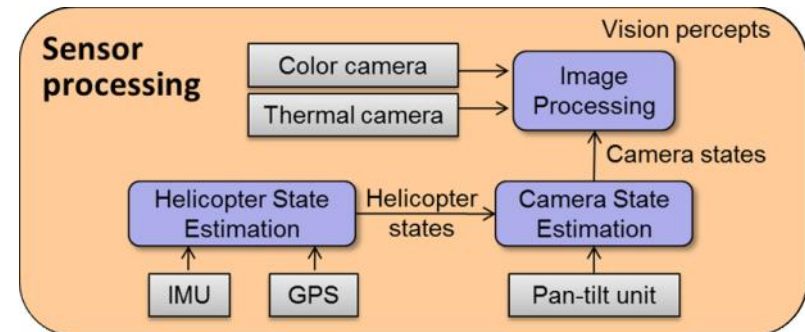
Grounding Stream Reasoning in Robotic Systems

[de Leng and Heintz FOFAI 2015, Heintz IROS 2014; de Leng and Heintz FUSION 2013]

- A temporal logical formula contains a number of symbols representing variables whose values over time must be collected and synchronized in order to determine the truth value of the formula.

forall x in UAV **always**(Speed[x] < 60)

Given a functional system, such as a robot, producing streams the grounding problem for logic-based stream reasoning is to *connect symbols in formulas to streams in the functional system so that the symbols get their intended meaning*.



Syntactic and Semantic Grounding

[de Leng and Heintz FOfAI 2015, Heintz IROS 2014; de Leng and Heintz FUSION 2013]

- *Syntactic grounding*: Use a direct mapping between symbols and streams, for example by using stream names in the formulas. A formula such as

forall x in UAV **always**(Speed [x] < 60)

would then have to be written something like

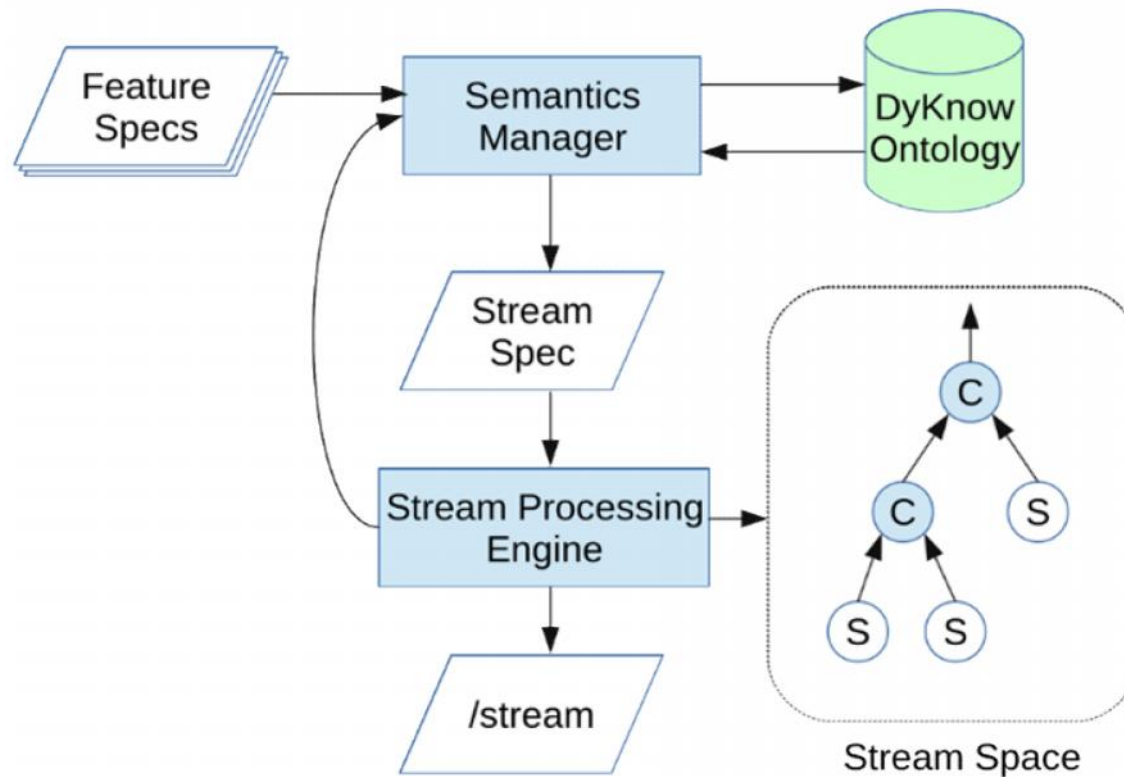
always((/uav1/uavstate.spd < 60) \wedge (/uav2/uavstate.spd < 60))

- *Semantic grounding*: Annotate streams with their semantic content and reason about how to connect symbols to streams using semantic web technologies. We call this reasoning for *semantic matching*. It finds the relevant streams by matching the ontological concepts used in a formula to the ontological concepts associated with the streams.



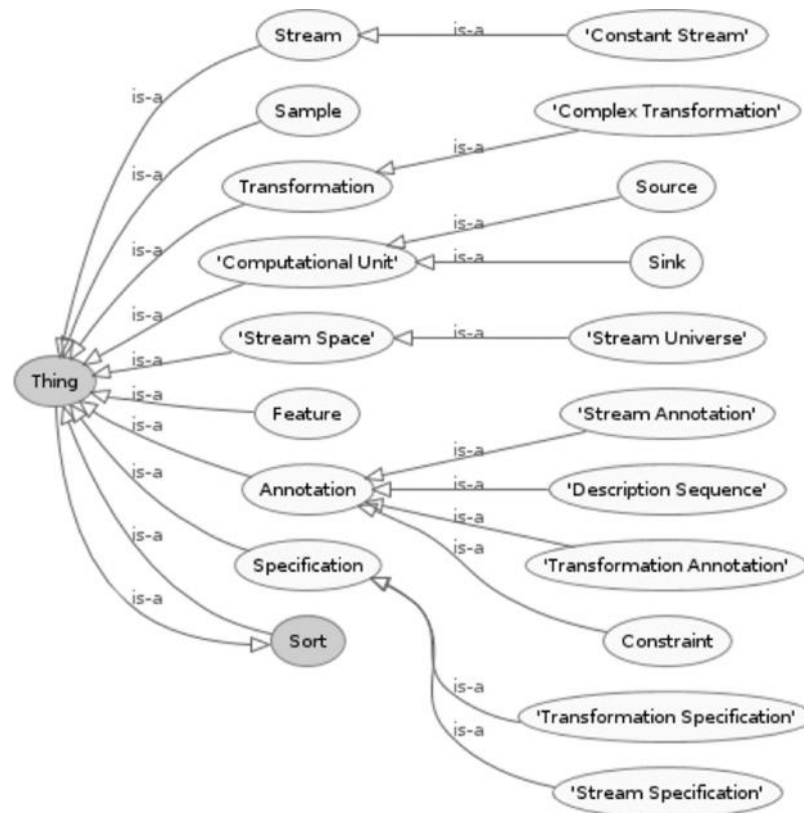
Semantically Grounded Stream Reasoning

[de Leng and Heintz FOFAI 2015, Heintz IROS 2014; de Leng and Heintz FUSION 2013]



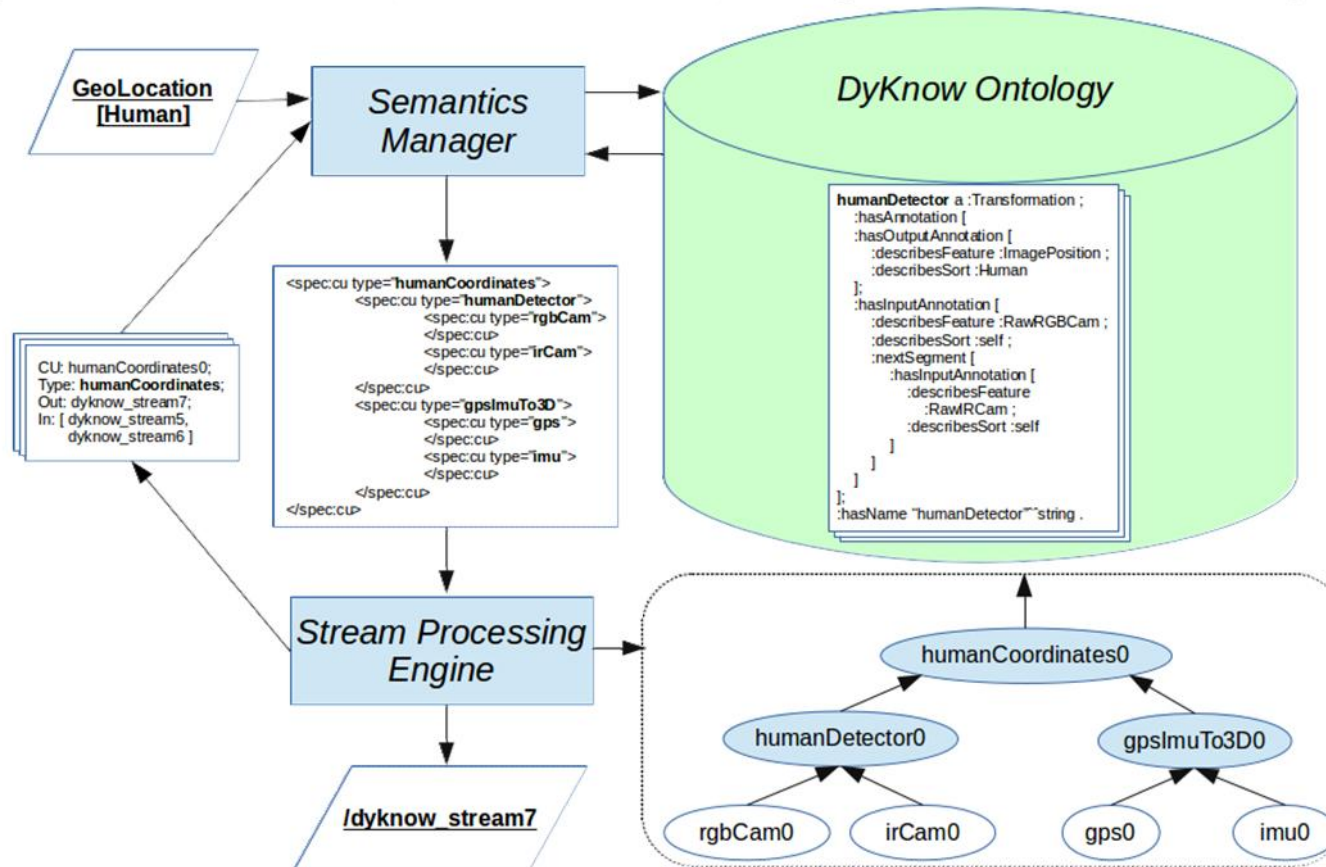
DyKnow Ontology

[deLeng and Heintz FOfAI 2015, Heintz IROS 2014; deLeng and Heintz FUSION 2013]



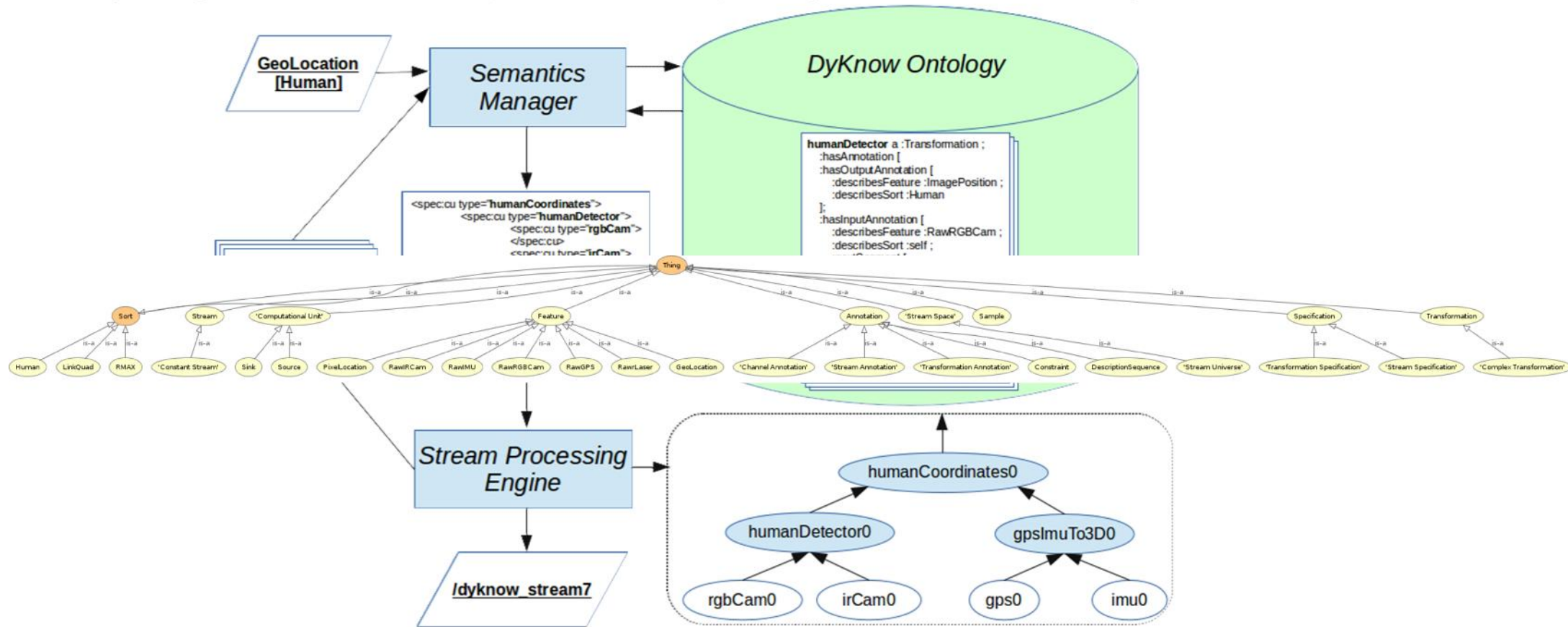
Semantically Grounded Stream Reasoning Example

[deLeng and Heintz FOFAI 2015, Heintz IROS 2014; deLeng and Heintz FUSION 2013]



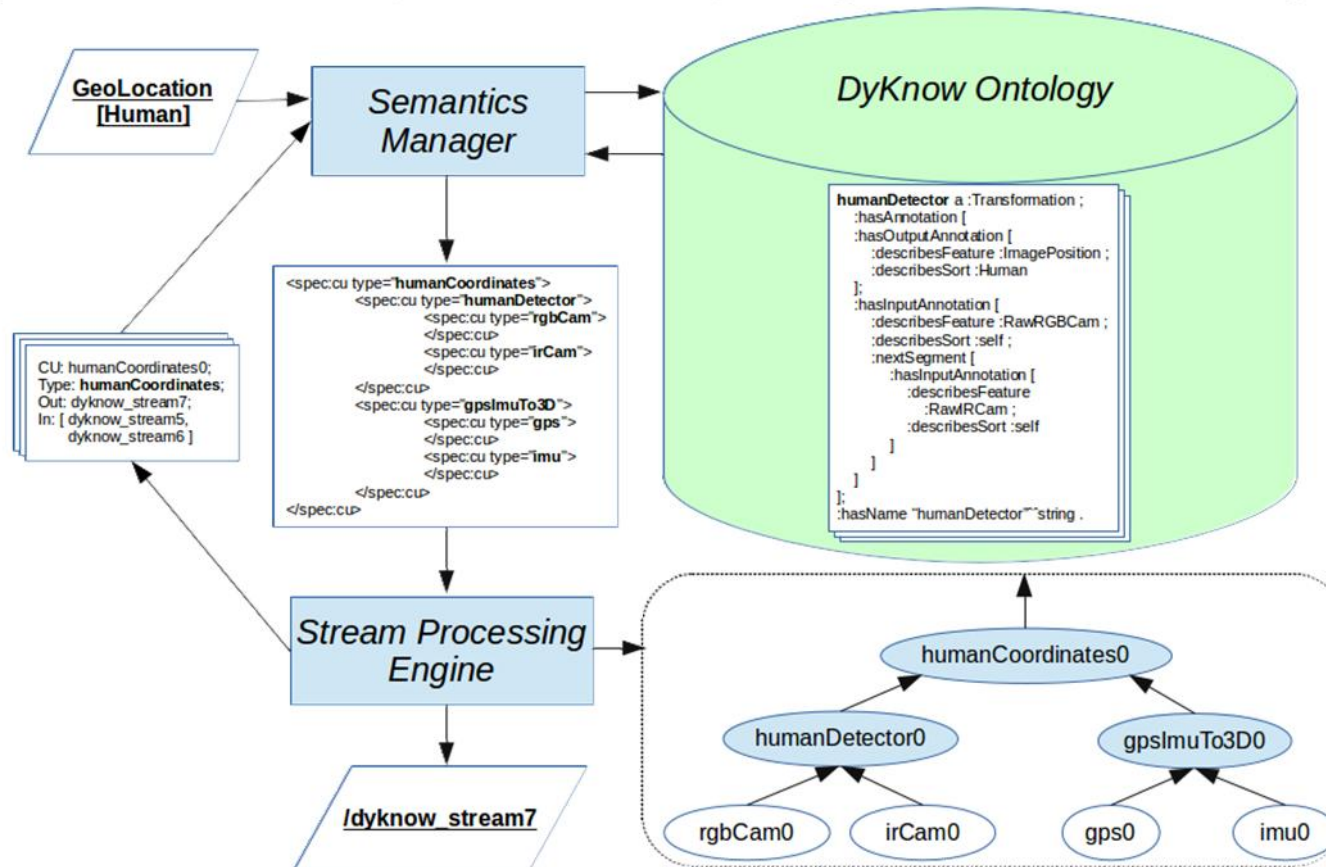
Semantically Grounded Stream Reasoning Example

[deLeng and Heintz FOFAI 2015, Heintz IROS 2014; deLeng and Heintz FUSION 2013]

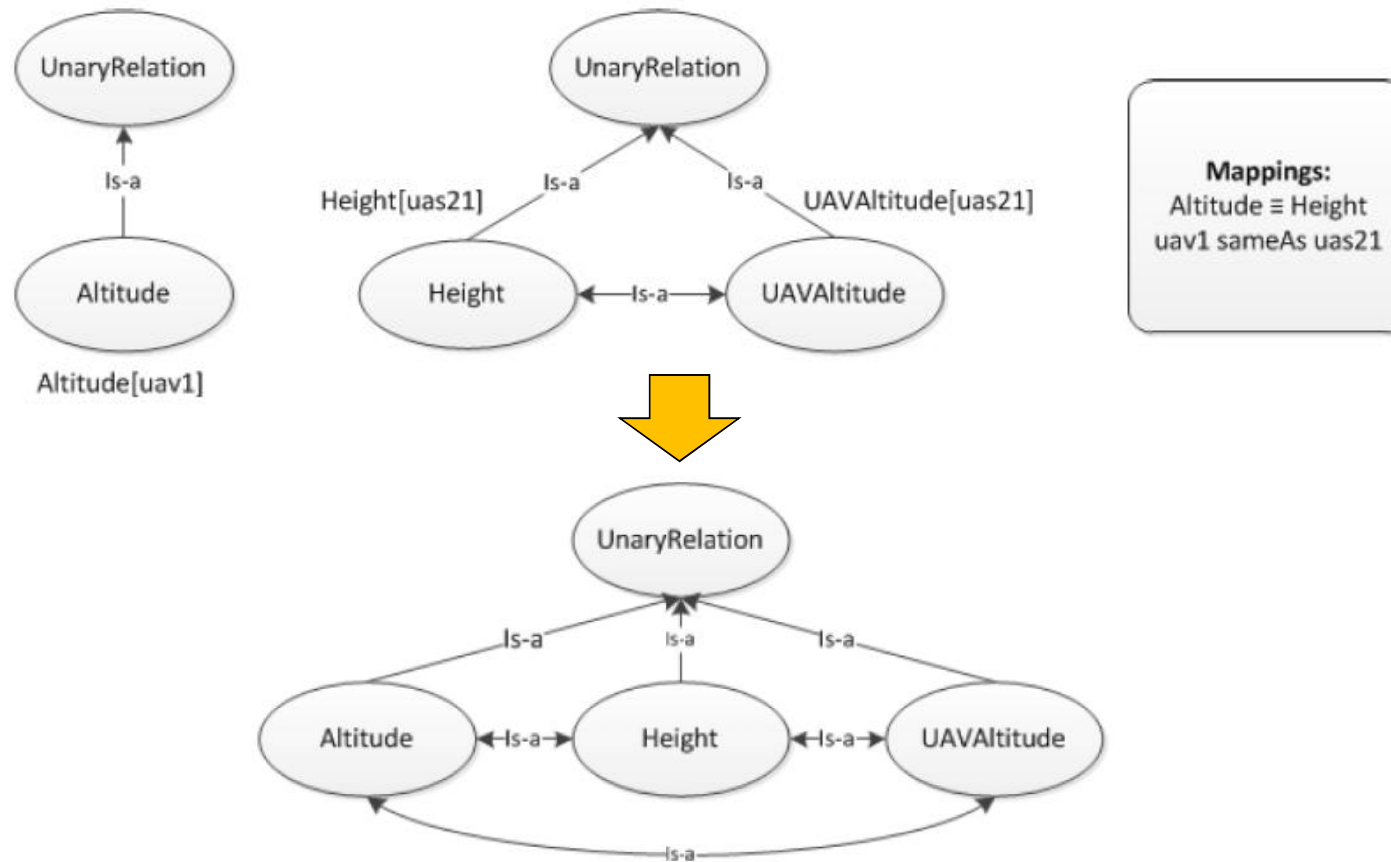


Semantically Grounded Stream Reasoning Example

[deLeng and Heintz FOFAI 2015, Heintz IROS 2014; deLeng and Heintz FUSION 2013]

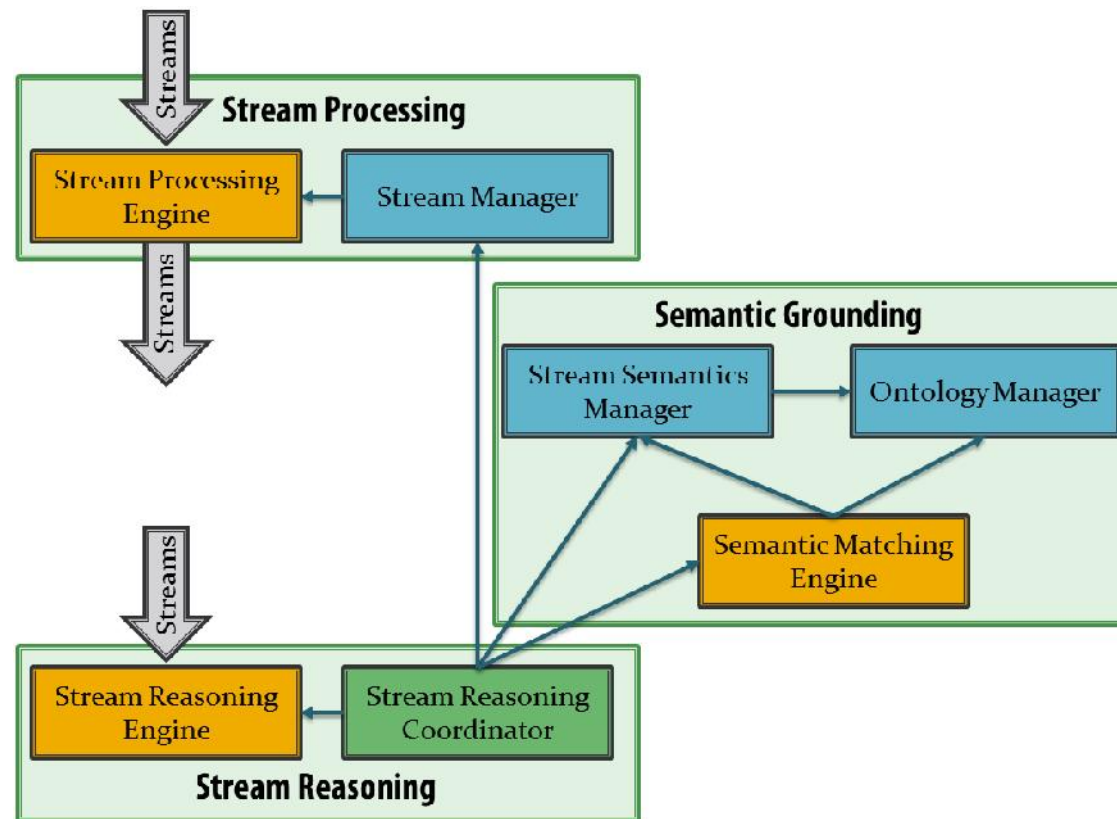


Semantic Matching with Multiple Robots



DyKnow Semantically Grounded Stream Reasoning in ROS

[Heintz IROS 2014]

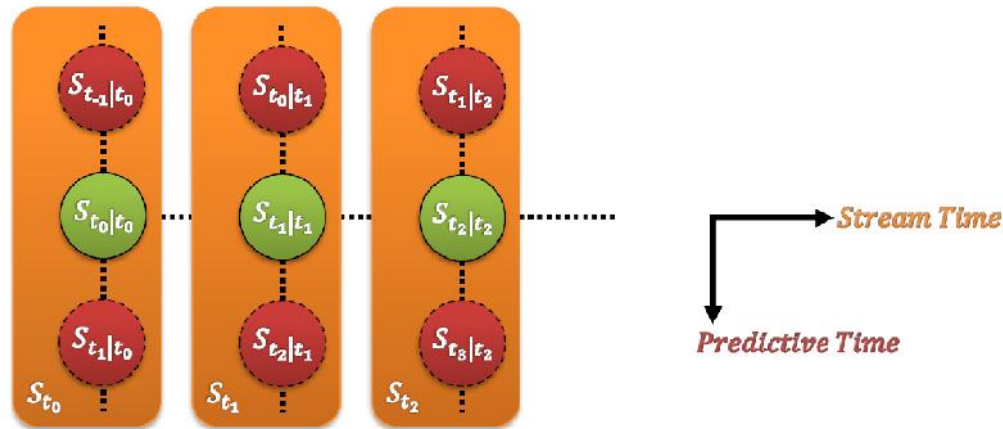


Future Work

Probabilistic Stream Reasoning

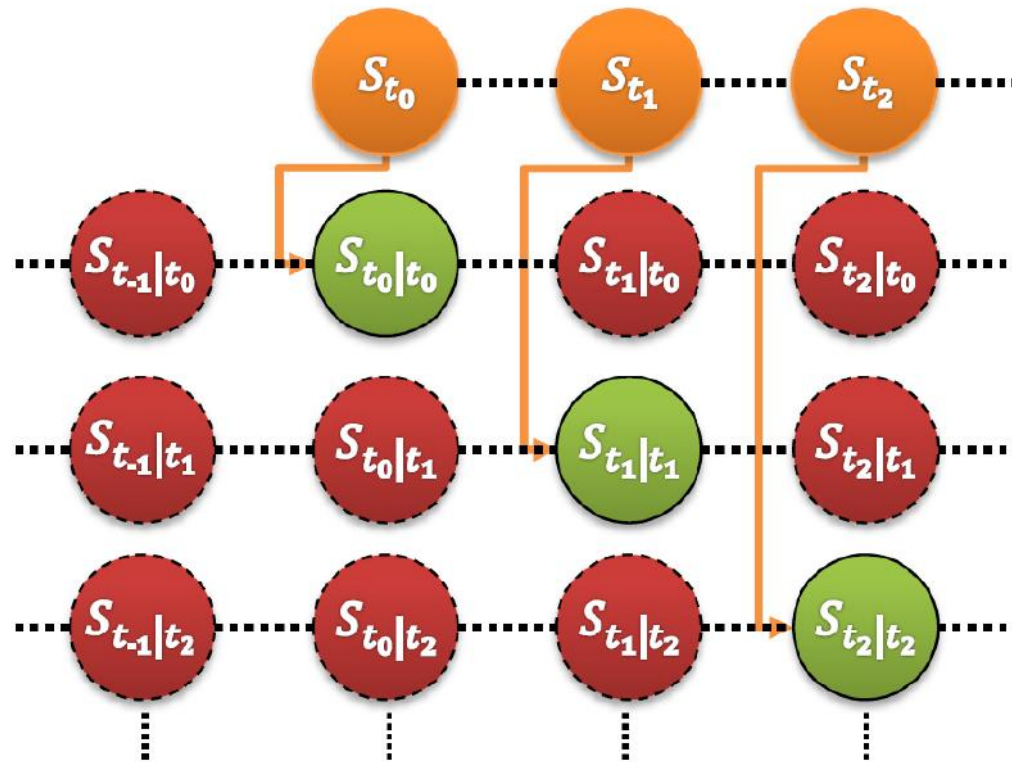
- Difficult to integrate logical and probabilistic reasoning. Robotics systems commonly do both separately in isolation.
- We are working on a formal interface between existing logical reasoning and existing probabilistic reasoning methods:
 - A formal framework with an explicit separation.
 - A selection of important temporal and probabilistic concepts from probability theory can be referred to at the logical level.
 - Both retain strengths and computational complexities.

Probabilistic Stream Reasoning



- Each state contains facts at a single time point
 - Truth values of predicates
 - Observed numerical values of terms
 - Stochastic estimates of terms (Green)
 - Stochastic predictions of terms (Red)

Probabilistic Stream Reasoning



Probabilistic Stream Reasoning

- MSTL is extended with stochastic terms and a special term operator:

Observed feature value: $\bullet_t \text{Altitude}[\mathbf{uav1}]$

Estimated feature value: $\bullet_{t|t} \text{Altitude}[\mathbf{uav1}]$

Predicted feature value: $\bullet_{t'|t} \text{Altitude}[\mathbf{uav1}]$

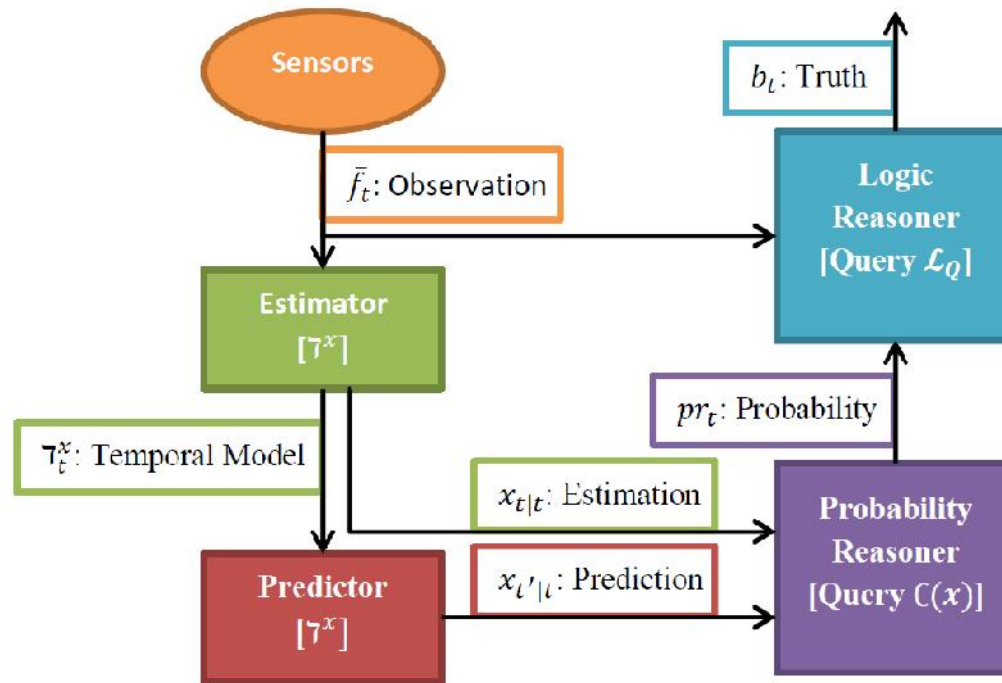
Example:

$$\Box (Pr((\text{Altitude}[\mathbf{roofA}] - \bullet_{0|0} \text{Altitude}[\mathbf{uav1}]) > 2m) \geq 0.99)$$

Probabilistic Stream Reasoning

- This allows us to express statements like:
 - Is my perception too uncertain?
 - Is my prediction too uncertain?
 - Does my prediction match my observation well enough?
 - Is my perception degrading?
 - Is my ability to predict degrading?

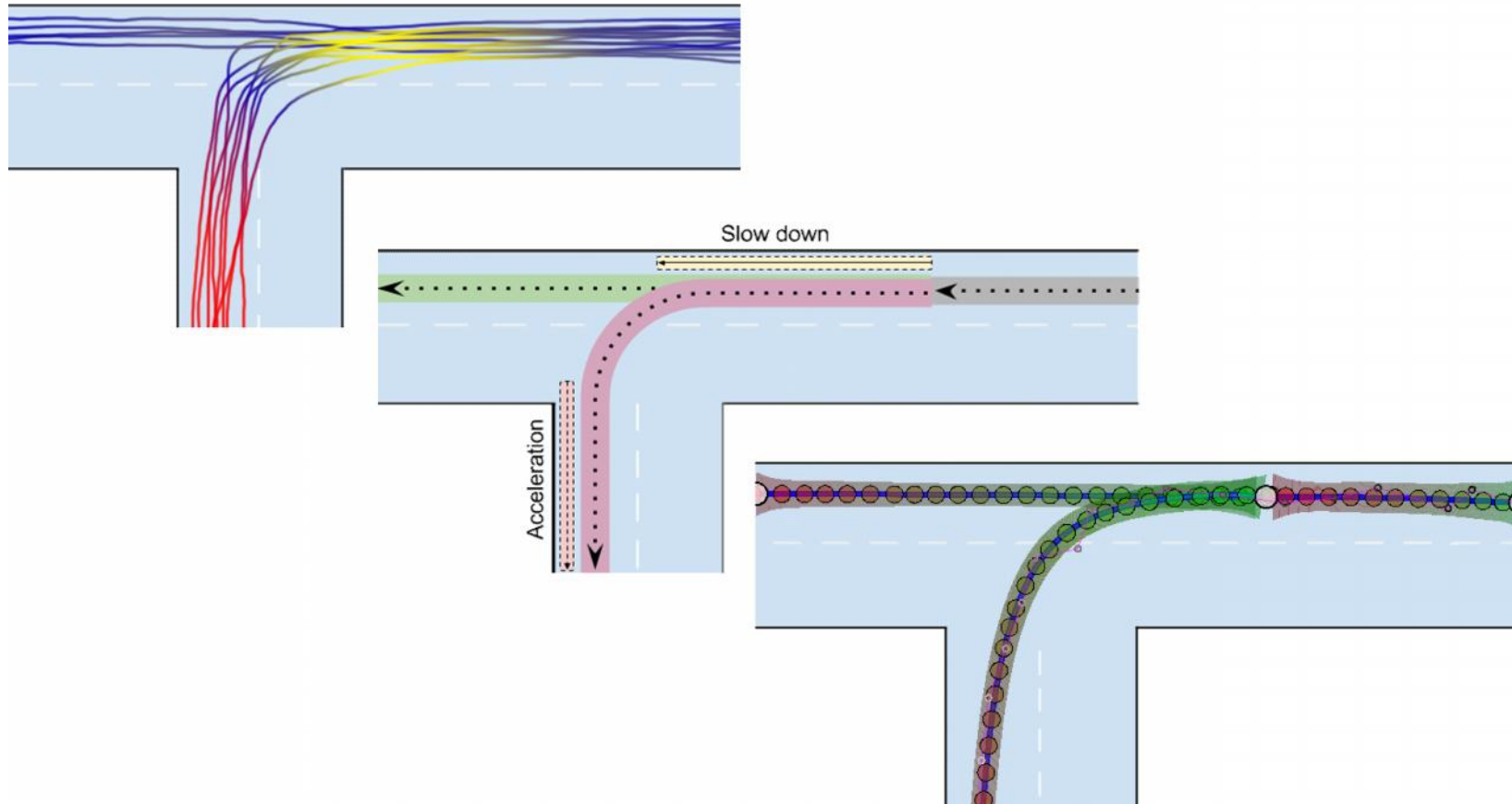
Probabilistic Stream Reasoning



$$\mathcal{E} = \langle T, \mathcal{O}, \mathcal{F}, \bar{\mathcal{F}}, \mathcal{X}, \mathcal{D}, \mathcal{T}, \mathcal{P} \rangle$$

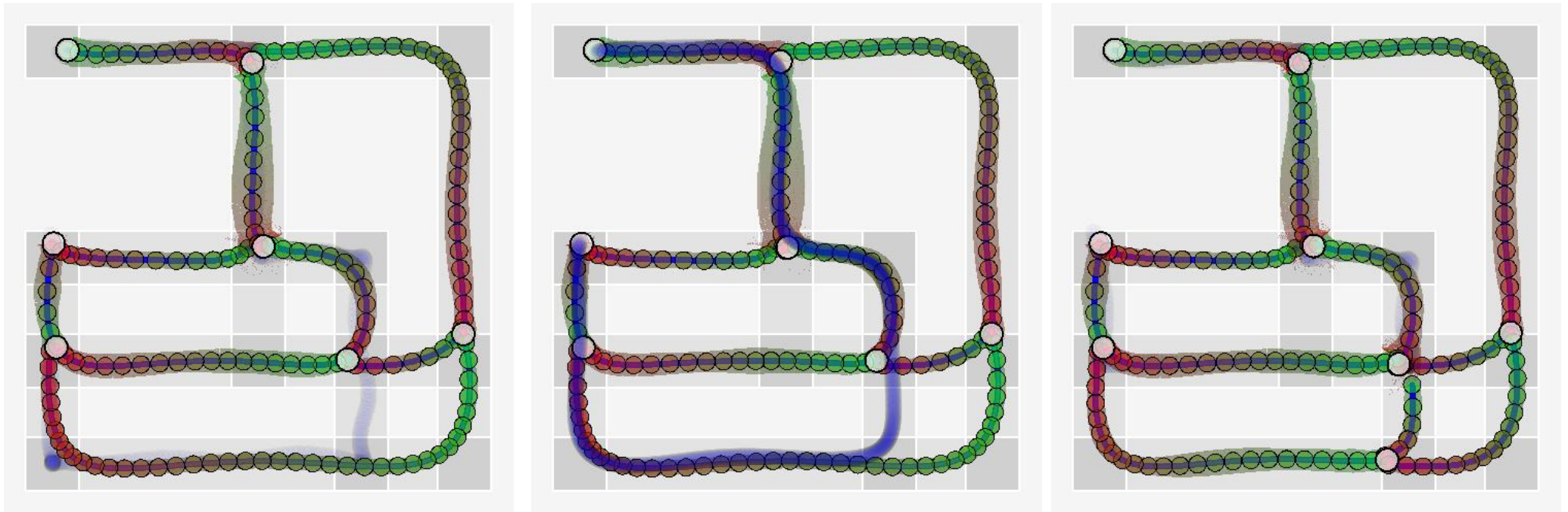
Unsupervised Learning of Activities

[Tiger and Heintz FUSION 2015, Tiger and Heintz STAIRS 2014]



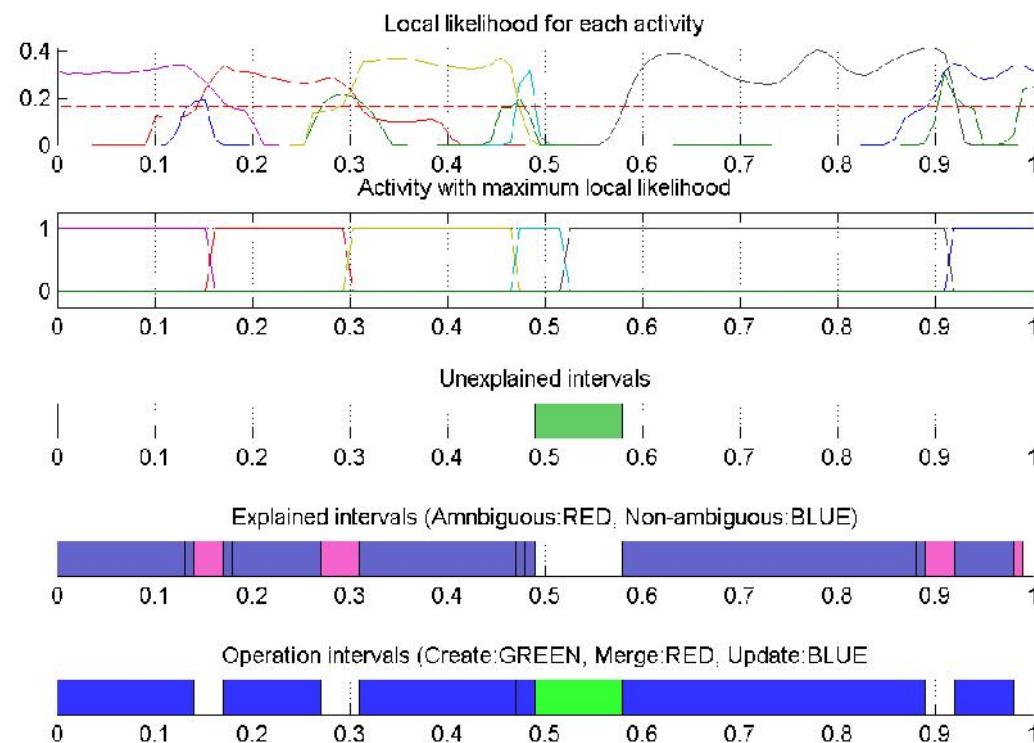
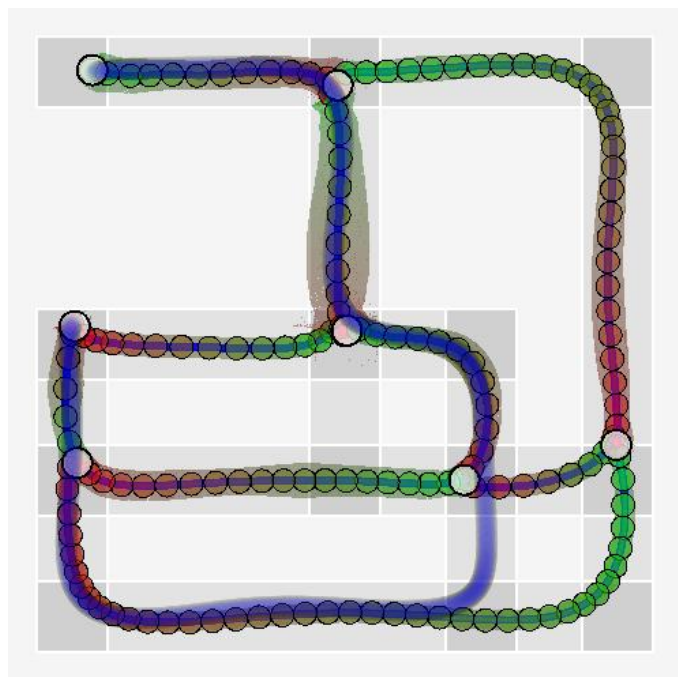
Unsupervised Learning of Activities

[Tiger and Heintz FUSION 2015, Tiger and Heintz STAIRS 2014]



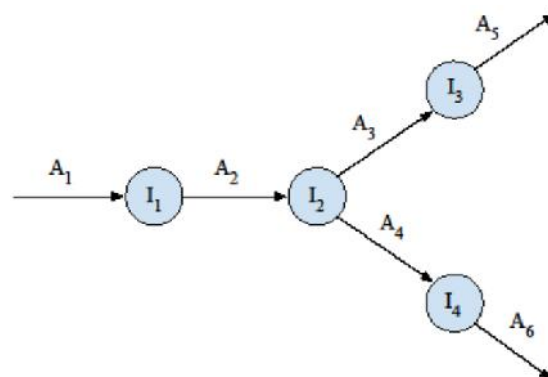
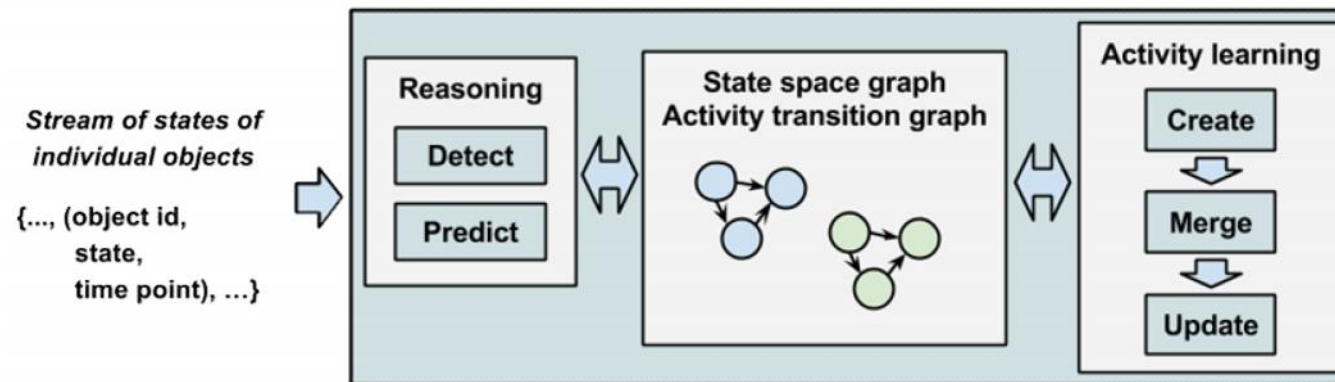
Activity Learning Example

[Tiger and Heintz FUSION 2015, Tiger and Heintz STAIRS 2014]

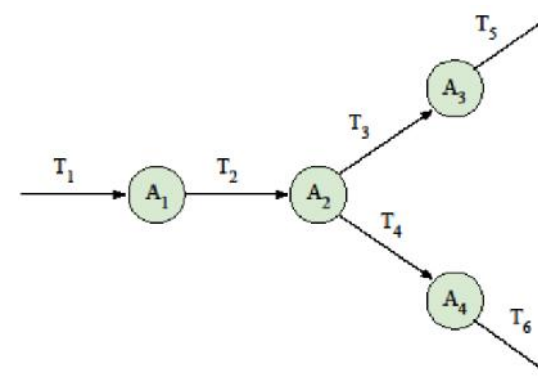


Unsupervised Learning of Activities

[Tiger and Heintz FUSION 2015, Tiger and Heintz STAIRS 2014]

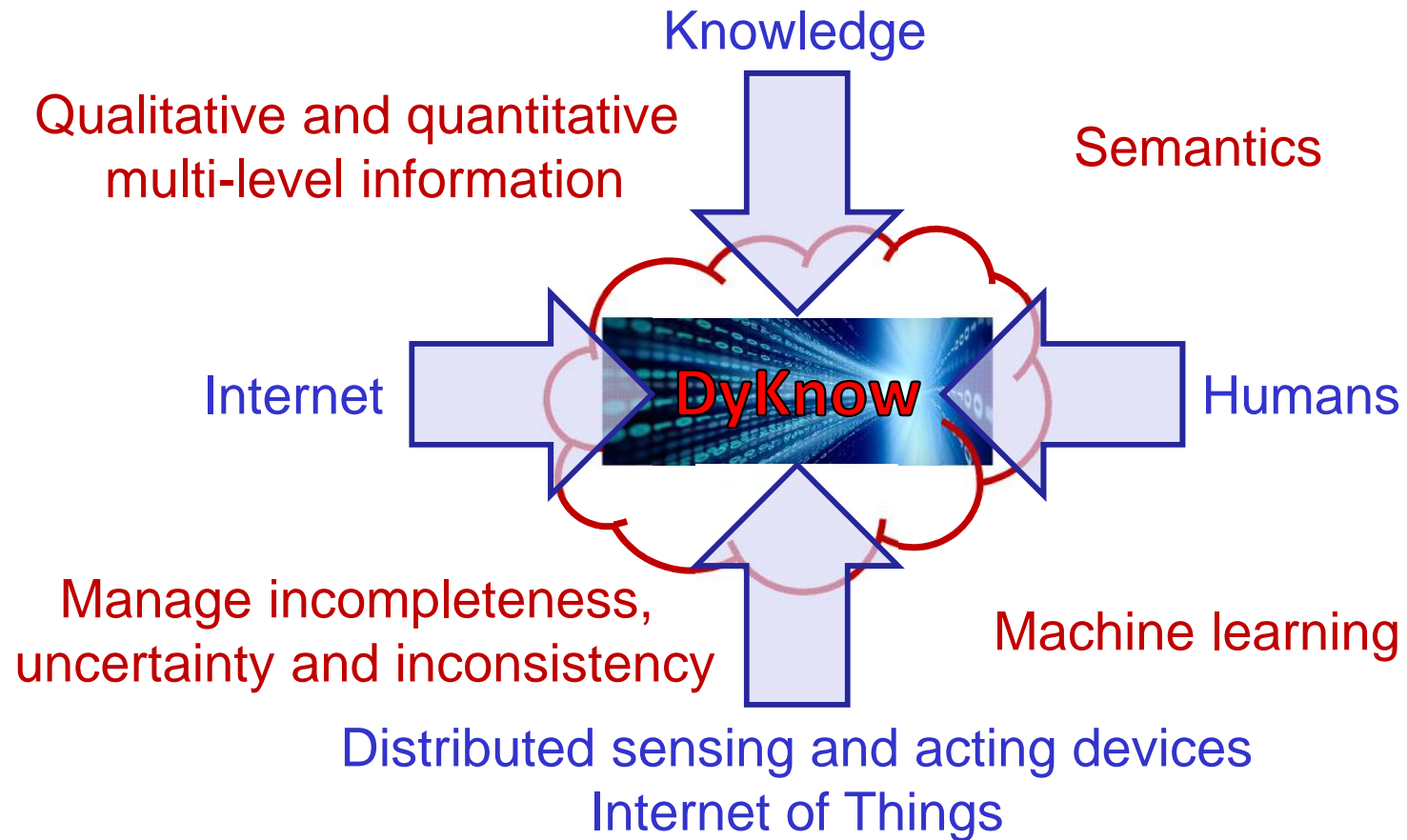


■ State space graph



■ Activity transition graph

Research Directions



Summary and Conclusions

Summary – Stream Reasoning

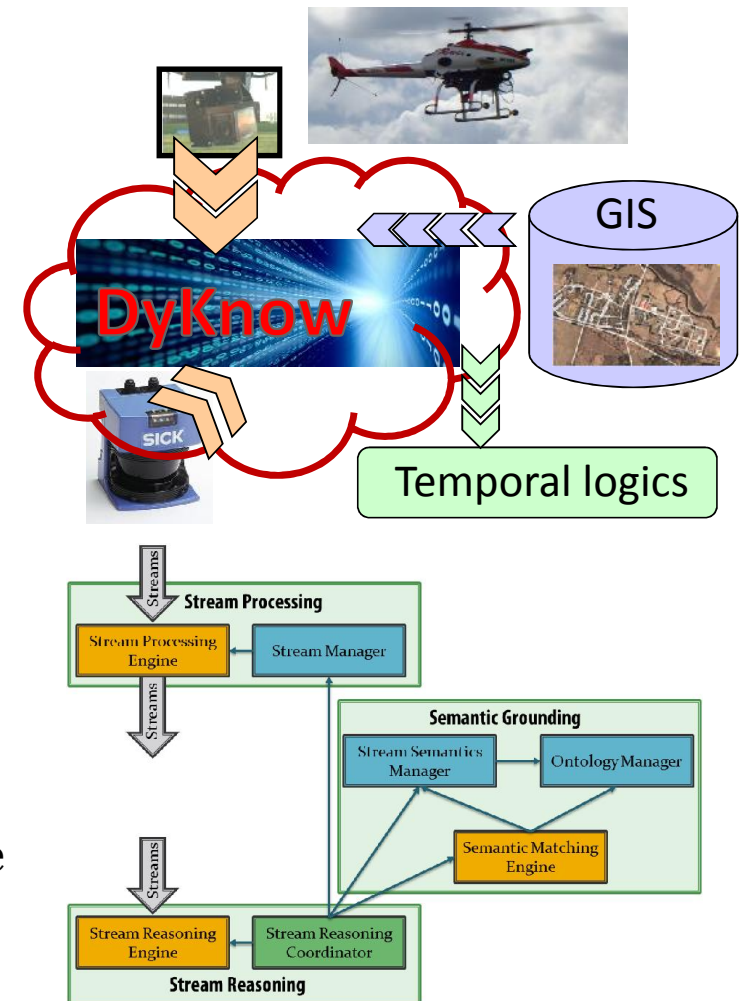
- **Streams** are potentially infinite sequences of elements.
- **Stream reasoning** is incremental reasoning over streams.
- Three different approaches:
 - **Window-based**, taking streams and turning them into relations over which standard reasoning/processing can be made. DSMS and RSP are both window-based.
 - **Event-based**, define and detect complex events in streams of events. Recognizing complex events ought to be a form of reasoning, it infers implicit information, which is a reasonable definition of reasoning.
 - **Logic-based**, evaluate temporal logical formulas over infinite streams.
- Discussion: Can they be unified to a single approach?

Summary – Spatio-Temporal Stream Reasoning

- The Metric Spatio-Temporal Logic MSTL combines MTL with RCC-8.
- MSTL formulas can be incrementally evaluated over streams of states through progression.
- MSTL supports spatial reasoning both within and between time-points.
- MSTL supports incomplete information through a three-valued logic approach.
- The logic-based approach is very suitable for making safe autonomous systems through for example execution monitoring with formal guarantees.

Conclusions

- High level incremental reasoning over streaming information is essential to autonomous systems for example to provide formal safety guarantees.
- DyKnow is a practical framework for grounded stream reasoning including support for spatio-temporal reasoning.
- The reasoning is semantically grounded through a common ontology and a specification of the semantic content of streams relative to the ontology.
- Through DyKnow, ROS is extended with a powerful stream reasoning capability available to a wide range of robotic systems.



Credits

- Slides:
 - Daniel de Leng, Emanuele Della Valle, Mattias Tiger
- Collaborators:
 - **Patrick Doherty, Jonas Kvarnström,**
 - **Daniel de Leng, Mattias Tiger,**
 - David Landén, Piotr Rudol, Mariusz Wzorek, Tommy Persson, Gianpaolo Conte, Karol Korwel, Zlatan Dragisic, Erik Frisk, Mattias Krysander, Jacob Roll, Alexander Kleiner and many others!
- Funding agencies: CADICS, CENIIT, CUGS, ELLIIT, EU, IDA, NFFP, SSF, Vinnova, VR

References to our Work

- Daniel de Leng and Fredrik Heintz. [Qualitative Spatio-Temporal Stream Reasoning With Unobservable Intertemporal Spatial Relations Using Landmarks](#). In Proc. AAAI 2016.
- Daniel de Leng and Fredrik Heintz. [Ontology-Based Introspection in Support of Stream Reasoning](#). Proc. SCAI 2015.
- Fredrik Heintz and Daniel de Leng. [Spatio-Temporal Stream Reasoning with Incomplete Spatial Information](#). In Proc. ECAI 2014.
- Daniel de Leng and Fredrik Heintz. [Towards On-Demand Semantic Event Processing for Stream Reasoning](#). In Proc. FUSION 2014.
- Fredrik Heintz. [Semantically Grounded Stream Reasoning Integrated with ROS](#). In Proc. IROS 2013.
- Fredrik Heintz and Daniel de Leng. [Semantic Information Integration with Transformations for Stream Reasoning](#). In Proc. FUSION 2013.
- Fredrik Heintz and Patrick Doherty. 2006. [A knowledge processing middleware framework and its relation to the JDL data fusion model](#). Journal of Intelligent & Fuzzy Systems, **17**(4):335–351. IOS Press.
- Fredrik Heintz and Patrick Doherty. 2004. [DyKnow: An approach to middleware for knowledge processing](#). Journal of Intelligent & Fuzzy Systems, **15**(1):3–13. IOS Press.
- Fredrik Heintz, Jonas Kvarnström and Patrick Doherty. [Stream-Based Hierarchical Anchoring](#). Künstliche Intelligenz, **27**(2):119–128. 2013. Springer. DOI: [10.1007/s13218-013-0239-2](#).
- Patrick Doherty, Jonas Kvarnström, Mariusz Wzorek, Piotr Rudol, Fredrik Heintz and Gianpaolo Conte. [HDRC3 - A Distributed Hybrid Deliberative/Reactive Architecture for Unmanned Aircraft Systems](#). In K. Valavanis, G. Vachtsevanos, editors, Handbook of Unmanned Aerial Vehicles, pages 849–952. Springer Science+Business Media B.V. DOI: [10.1007/978-90-481-9707-1_118](#).
- Fredrik Heintz, Jonas Kvarnström and Patrick Doherty. [Stream-Based Reasoning Support for Autonomous Systems](#). In Proc. ECAI 2010.
- Mattias Krysander, Fredrik Heintz, Jacob Roll and Erik Frisk. [FlexDx: A Reconfigurable Diagnosis Framework](#). Engineering applications of artificial intelligence, **23**(8):1303–1313. 2010. Elsevier. DOI: [10.1016/j.engappai.2010.01.004](#).
- Fredrik Heintz, Jonas Kvarnström and Patrick Doherty. [Bridging the sense-reasoning gap: DyKnow - Stream-based middleware for knowledge processing](#). Advanced Engineering Informatics, **24**(1):14–26. 2010. Elsevier. DOI: [10.1016/j.aei.2009.08.007](#).
- Patrick Doherty, Jonas Kvarnström and Fredrik Heintz. [A Temporal Logic-based Planning and Execution Monitoring Framework for Unmanned Aircraft Systems](#). Autonomous Agents and Multi-Agent Systems, **19**(3):332–377. 2009. Springer. DOI: [10.1007/s10458-009-9079-8](#).