[dstl] The Science Inside

Stone Soup open source framework: applications to decentralised tracking

Jordi Barr

Dstl Porton Down

Ministry of Defence

18 January 2022

(based on DSTL/PUB126038)

Contents

[dstl] The Science Inside

- (Distributed) tracking and state estimation
- Open source
- Stone Soup

- A distributed sensing example
- Applications

- What next?
- Impact and implications

Acknowledgements

- *Steve Hiscocks, Paul Thomas, Rich Green, James Wright, Oliver Harrald, Sebastian Vidal, Nikki Perree, Henry Pritchett, Daniel Cherrie, Ed Rogers and Jonathan Osborne (Dstl)*
- *Simon Maskell, Lyudmil Vladimirov (University of Liverpool)*

Tracking and state estimation

- **Critical for military Situational Awareness**






Distributed tracking and state estimation

dstl The Science Inside






Open Source[1]

Image © Paul Thomas. Used with permission

The ‘Cathedral’ approach

- *“…built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time.”*

Image © Paul Thomas. Used with permission

The ‘Bazaar’ approach

- *“Release early and often, delegate everything you can, be open to the point of promiscuity … seemed to resemble a great babbling bazaar of differing agendas and approaches.”*

[1] from The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. Eric S. Raymond 1998

Differing, perhaps conflicting, requirements

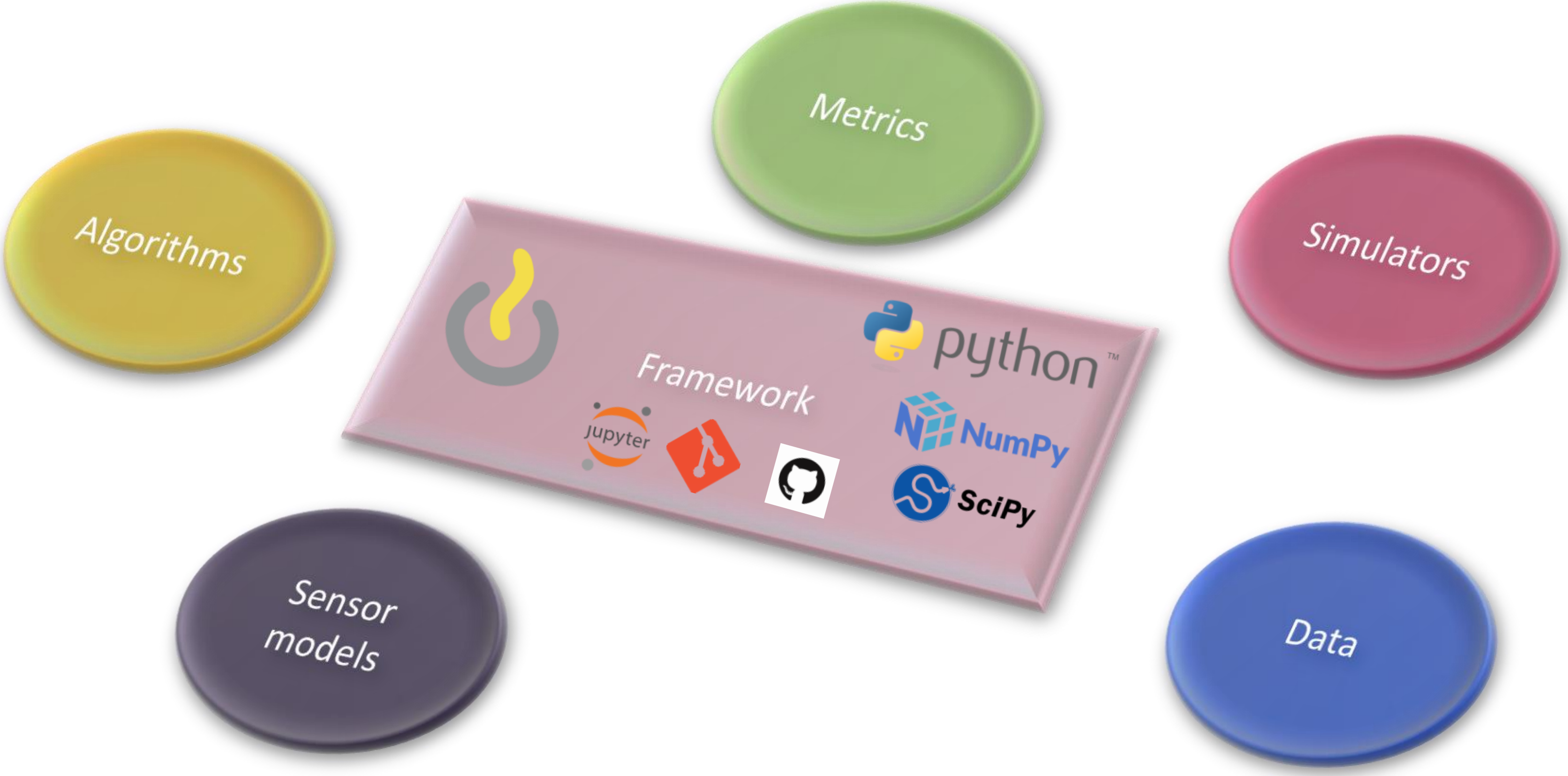
[dstl] The Science Inside

- **Academic researcher:** **“***I wish I could show the performance improvement of my new algorithm without having to reproduce others’ methods.*”
- **Defence industry:** “*I wish I could use state-of-the-art approaches in the new product I’m developing.*”
- **Government Laboratory:** “*At conferences I hear about so many new methods. I wish I knew which of these actually works best in practice*.”

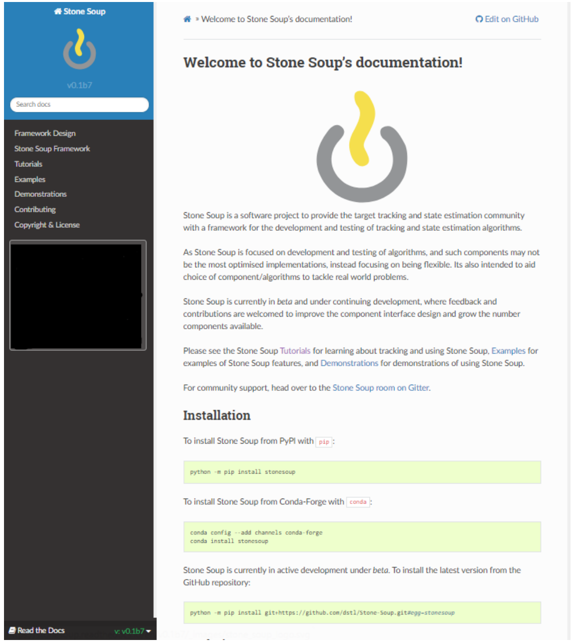
Stone Soup

The Science Inside

- **The open source tracking and state-estimation framework**



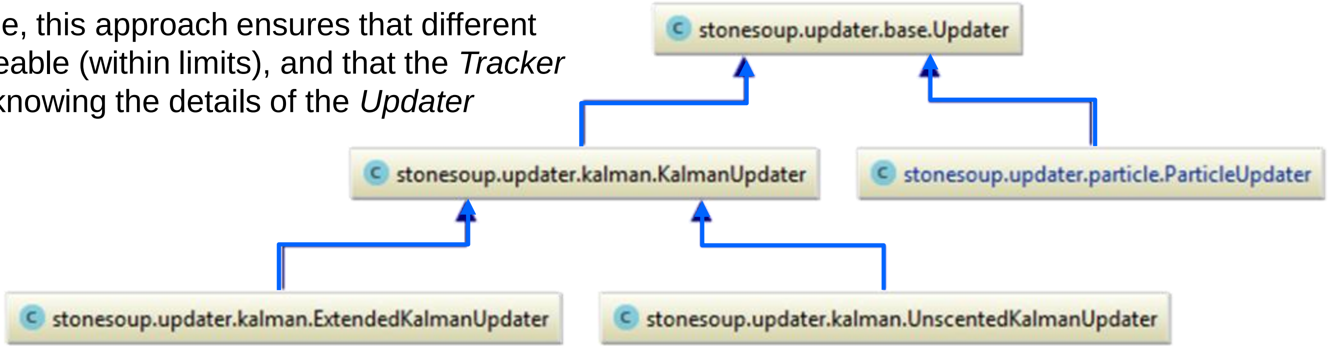
- **Stone Soup's key design principles**
  - Open source
    - Members of the tracking/state estimation community can contribute enhancements
  - Modular
    - Trackers are formed by assembling components in a pre-defined way
  - Interchangeable
    - Component classes have inheritance from base classes so that components of the same type have identical interfaces
  - Well documented
    - Documentation available on readthedocs.org[2]



[2] https://stonesoup.readthedocs.io/en/latest/index.html

Stone Soup is object oriented

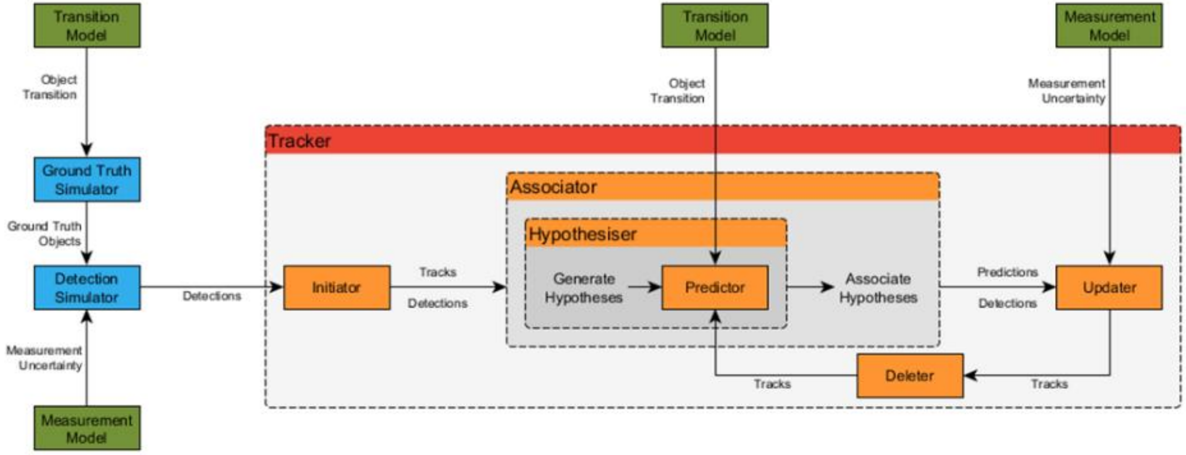
- **Abstraction**
  - Stone Soup trackers are built as hierarchical objects. For example, a *MultiTargetTracker* object may contain track *Initiator*, a track *Deleter*, *Detector*, *DataAssociator*, and *Updater* objects. Each of these objects is defined by an abstract class that specifies the external interface for that class.
- **Inheritance**
  - An example, the *Updater* abstract class specifies that an *Updater* object must have a measurement_model attribute, and that it must have methods *predict_measurement()* and *update()* that returns a *MeasurementPrediction* and *State* object respectively. Therefore, all implementations of Updaters in Stone Soup (*KalmanUpdater, ExtendedKalmanUpdater*, *ParticleUpdater,* etc.) must have the specified elements.
- **Encapsulation**
  - With the *Updater* example, this approach ensures that different Updaters are interchangeable (within limits), and that the *Tracker* can utilize them without knowing the details of the *Updater* implementation.



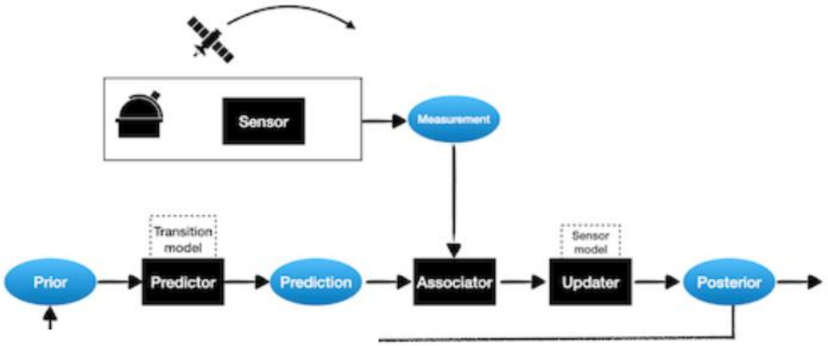
Structure is key

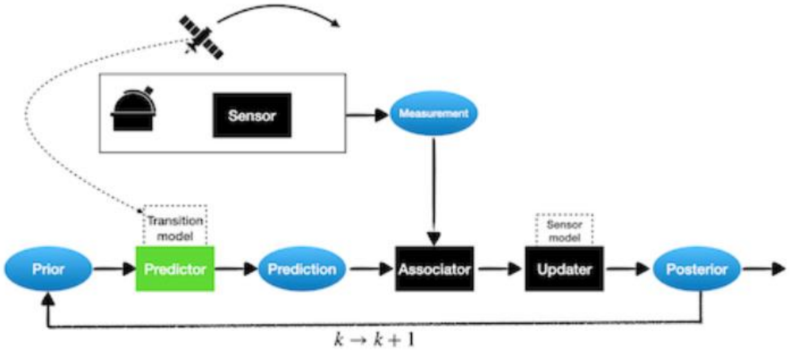
Algorithm sub-structure

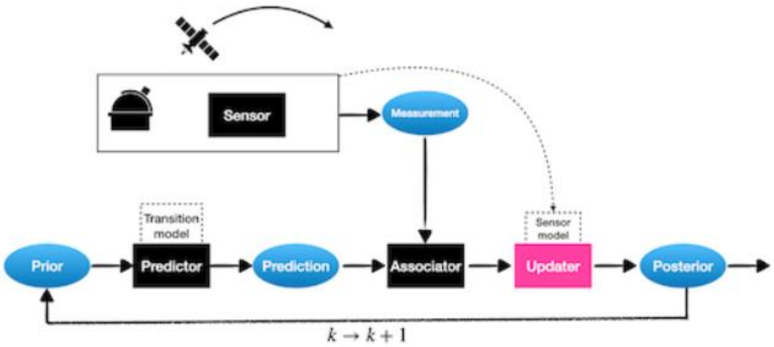
- **Core classes**
  - Transition Model
  - Measurement Model
  - Initiator
  - Deleter
  - Associator
  - Hypothesiser
  - Predictor
  - Updater



Tracking as a machine

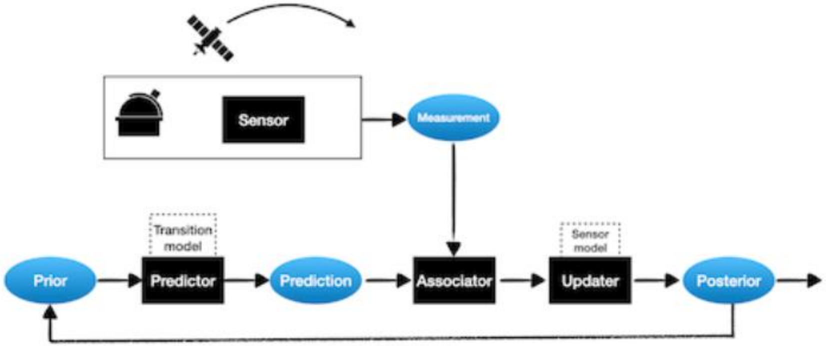


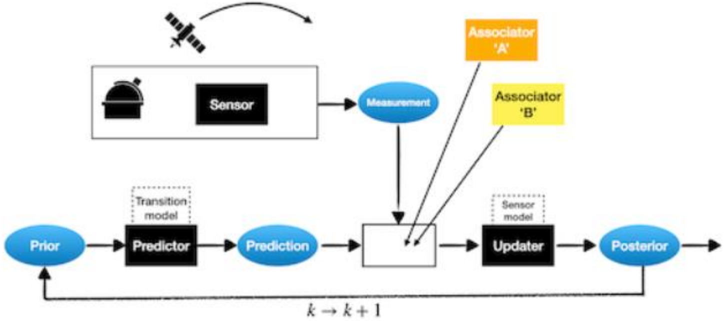


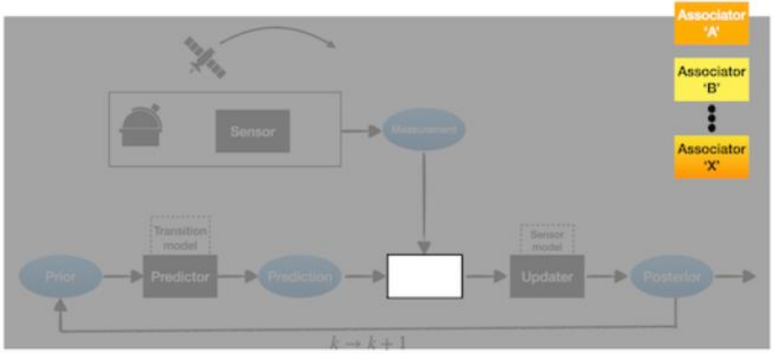


Tracking as a machine

[dstl] The Science Inside







Algorithms available

Filters (predictor + updater)

- Kalman filter family
  - Including KF, EKF, UKF, √Kalman, Iterated Kalman.
- Particle filter family
- Gaussian mixture point process family
  - Including GM-PHD, CPHD, LCC

Deleters

- Covariance-based
- Time-based
- Composite

Hypothesisers

- Distance
- PDA
- Gaussian Mixture

Gaters

- Distance-based
- Filtered detections

Associators

- Nearest neighbour
- Global nearest neighbour
- 2D assignment
- Probabilistic data association (PDA)
- Joint probabilistic data association (JPDA)
- Track to track
- Track to truth

Initiators

- Gaussian
- Particle
- Single point
- Simple measurement
- Multiple measurement
- Gaussian particle

Mixture reducers

- Gaussian mixture reducer

Metrics

- OSPA/GOSPA metrics
- SIAP metrics

Simulators available

Simulators

- Single target
- Single target, multi-transition model
- Multiple target
- Multi-target, multi-transition model
- Simple detection simulator
- Simple detection, multi-transition model
- Platform detection simulator

Sensors

- Base sensor
- Passive bearing/elevation sensor
- Bearing/range radar
- Bearing/range rotating radar
- Elevation/bearing/range radar
- Bearing/range-rate radar
- Elevation/bearing/range-rate radar
- Bearing/range raster-scan radar
- AESA radar

Platforms

- Fixed
- Moving
- Moving, multi-transition model

Radar beam patterns/shapes

- Beam transition model
- Stationary beam pattern
- Beam sweep pattern
- 2D Gaussian shape

Decentralised playground

[dstl] The Science Inside

Simple examples of decentralised fusion using Stone Soup

In [1]:
```
%matplotlib notebook
```

This notebook is intended to show some basic examples of how to begin do decentralised data fusion in Stone Soup. A basic single-target, multi-sensor example is used to demonstrate different ways of passing tracks and detections between sensors and forming different views on the global situational awareness.

A small number of basic elements, a predictor, hypothesiser and updater are used over and again. The differences come in whether messages are passed to a central location, or between fusion centres, and also whether passed as detections or tracks.

Start by setting the time

In [2]:
```
from datetime import datetime, timedelta
start_time = datetime.now()
```

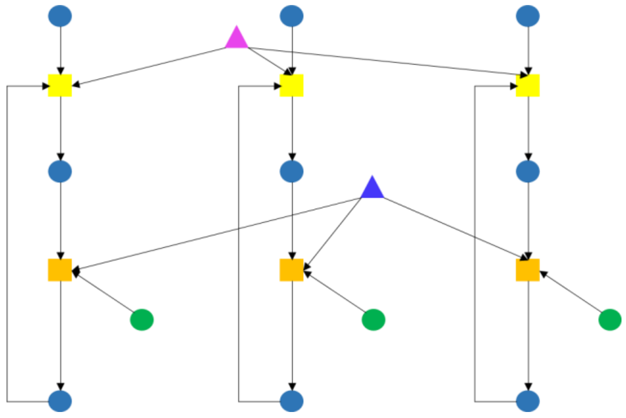
Ground truth

Simulate a target moving due east on the 2d x-y plane at near constant velocity. Period of the simulation is 200 time steps and the average speed is 1 unit/time step.

In [3]:
```
from stonesoup.types.groundtruth import GroundTruthPath, GroundTruthState
from stonesoup.models.transition.linear import CombinedLinearGaussianTransitionModel, \
                                               ConstantVelocity

## Set the x and y noise parameters separately.
q_x = 0.00005
q_y = 0.00005
transition_model = CombinedLinearGaussianTransitionModel([ConstantVelocity(q_x),
                                                          ConstantVelocity(q_y)])

truth = GroundTruthPath([GroundTruthState([0, 1, 0, 0], timestamp=start_time)])

num_steps = 200
for k in range(1, num_steps + 1):
    truth.append(GroundTruthState(
        transition_model.function(truth[k-1], noise=True, time_interval=timedelta(seconds=1)),
        timestamp=start_time+timedelta(seconds=k)))
```

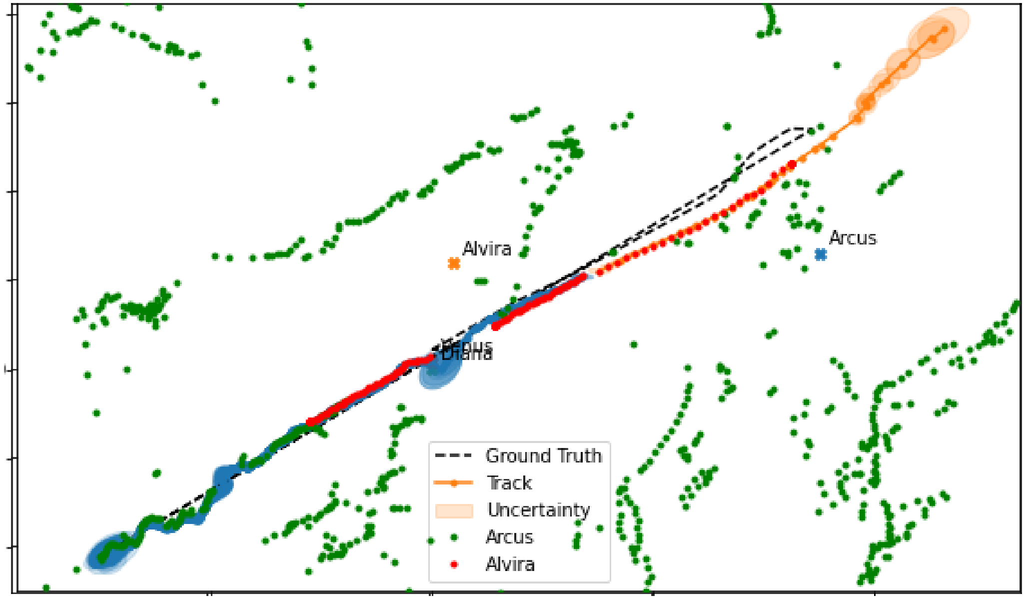
Sensors

Place three range-bearing sensors at (50,-50), (100, 50), (150,-50), each with a 360 degree field of view. We give them a probability of detection of 0.1 (i.e.



Counter UAS – Kaggle data challenge

- Real-time tracking, detection fusion, giving estimate of location, only using data up to current time.
- Out the box Stone Soup:
  - Utilised a extended Kalman filter using a (near) constant velocity model
  - Used Mahalanobis distance from location estimates to detections, with global nearest neighbour association.
  - Multi-point track initialisation, and time/uncertainty based deletion.
- Custom components:
  - Track initiation only based on range-bearing radar sensors, where identification was (suspected) drone.
  - Data association, avoid using sensor data when track estimate close.



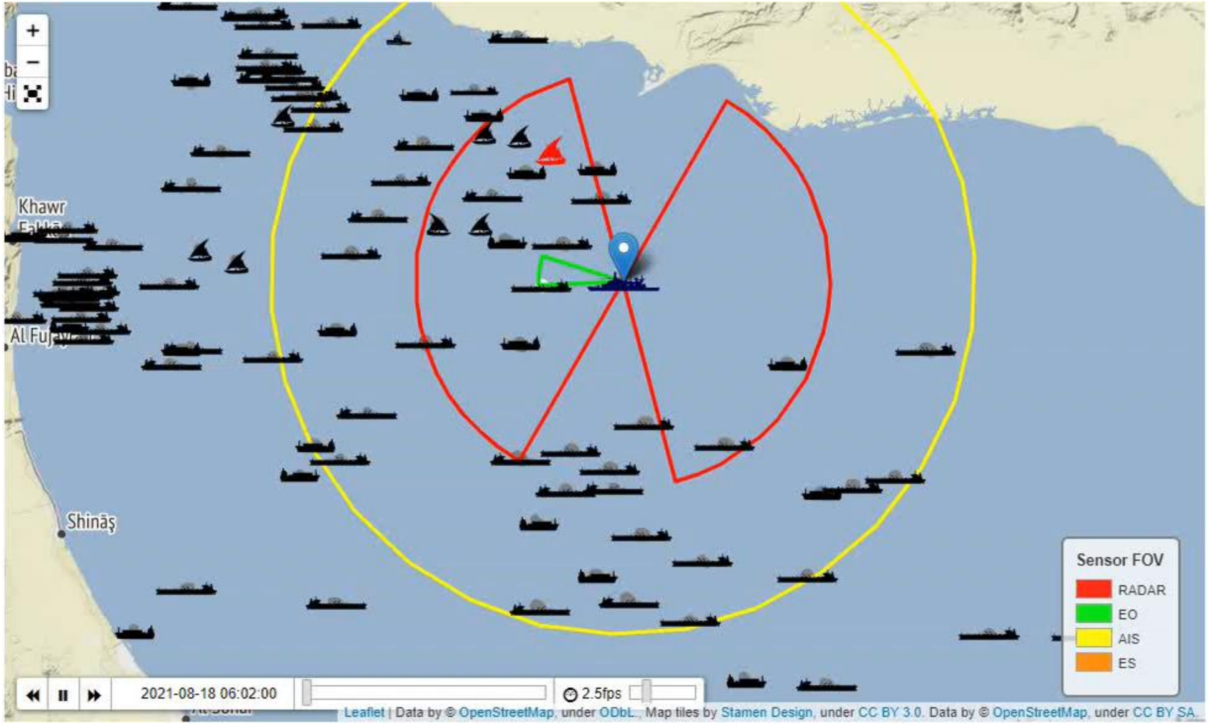
Sensor management

[dstl] The Science Inside

- **Maritime multi-target, multi-sensor, multi-platform**

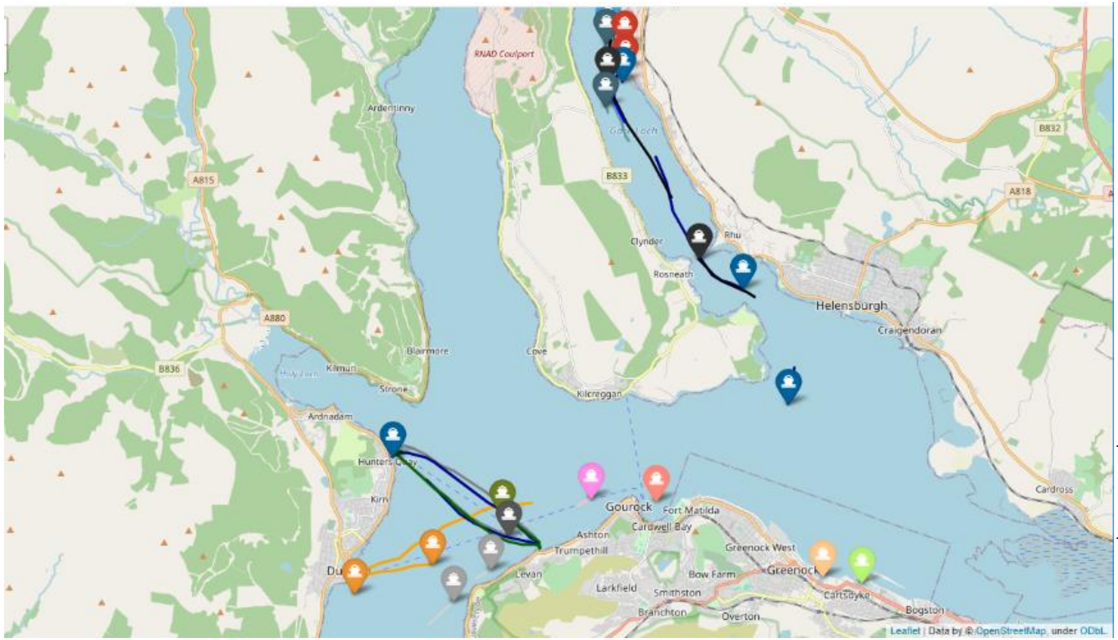
Intro tutorial: https://gist.github.com/nperree-dstl/6eed33102d80805a6142cc4218b4185f
Multiple sensor management: https://gist.github.com/nperree-dstl/61a72367b1c9e3396e05f9439956022f
Target based sensor management: https://gist.github.com/nperree-dstl/f352e7026a243183bbe0a2fd584b9906
Demo of all methods available in Stone Soup: https://gist.github.com/nperree-dstl/8ec887b98fa909b02e79715034bac94c
GitHub: https://github.com/dstl/Stone-Soup/pull/503

- **Example courtesy of Nikki Perree, Henry Pritchett (Dstl)**



Multi modal fusion for tracking

- **Radar and AIS data**
  - Predictor and Updater = unscented Kalman
  - Associator = Global nearest neighbour
  - Transition model = Ornstein Uhlenbeck



- **Courtesy of Steve Hiscocks (Dstl)**

Examples of use

Multiple vehicle tracking

- **Using *TensorFlowBoxObjectDetector* class as the detector**
  - Filtering on objects of type ‘car’ with a detection score > 0.1
  - Uses *FrameReader* class
  - Uses a *MultiTargetTracker* based on *KalmanPredictor* and *Kalman Updater*
  - Uses *GNNWith2DAssignment* class for data association
  - *ConstantVelocity* model for bounding box location
  - *RandomWalk* model for bounding box size

- **Example courtesy of Lyudmil Vladimirov (University of Liverpool)**
- **Original data Karol Majek https://www.youtube.com/watch?v=MNn9qKG2UFI**

Examples of use

[dstl] The Science Inside

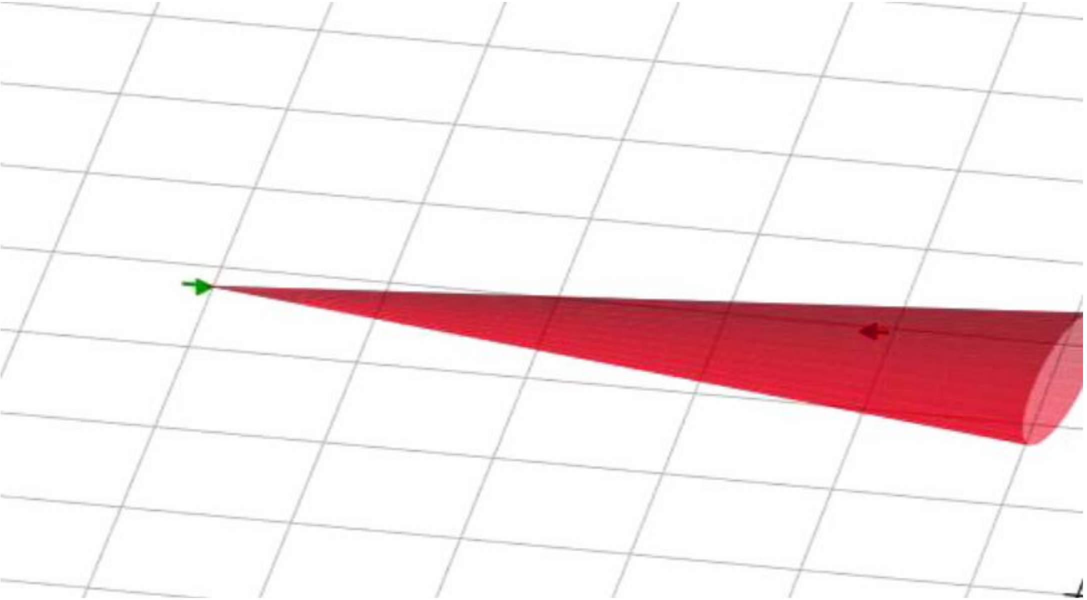
People tracking in congested areas

- **MOT20 Challenge data**
  - Using a Tensorflow DNN as a detector
  - Vanilla Kalman filter
  - Constant acceleration model for bounding box location
  - Constant velocity model for bounding box size

- **Example courtesy of Steve Hiscocks (Dstl)**
- **Original data motchallenge.net**

Stress-testing of operational tracking systems

- **Typhoon radar modelling**
  - Air intercept radar modelling
  - two jets manoeuvring as they close
  - the radar beam of the green jet shown by the red cone, and detections of the red jet marked by blue dots



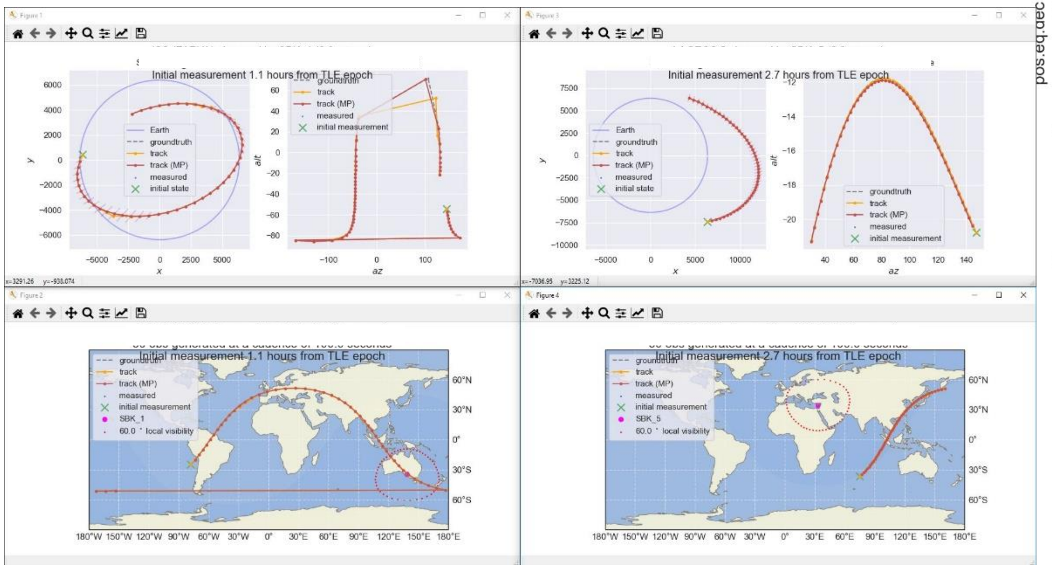
- Note: synthetic parameters were used to avoid classification restrictions

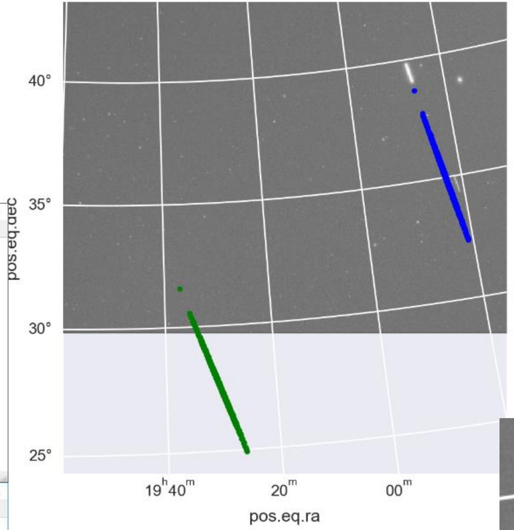
- **Example courtesy of Ed Rogers and Jonathan Osborne (Dstl)**

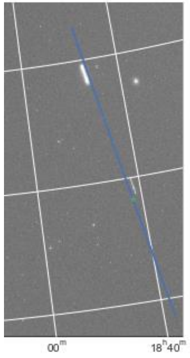
Examples of use

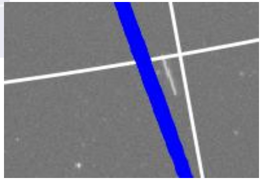
Space Surveillance

- **Uncertainty in orbit track estimation**
- **Comparing dynamical models**









Examples of use

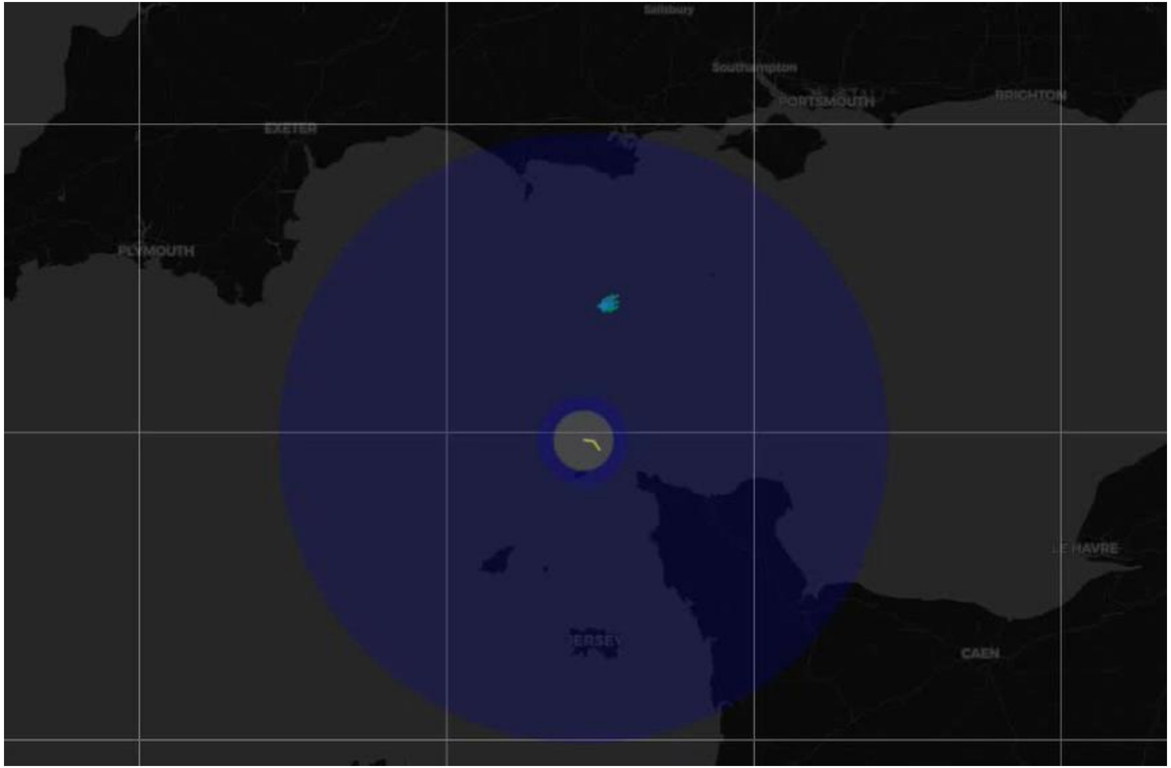
dstl The Science Inside

Advanced scenario generation

- **Maritime Situational Awareness**
  - Sensors:
    - RN vessel with Elint and radar.
    - Ship's helicopter with radar.
    - Occasional overhead sensing.
  - Activity:
    - Drugs trade ship moving alongside group of dhows
    - Two split off from the main pack
    - Ship's helicopter sent to intercept

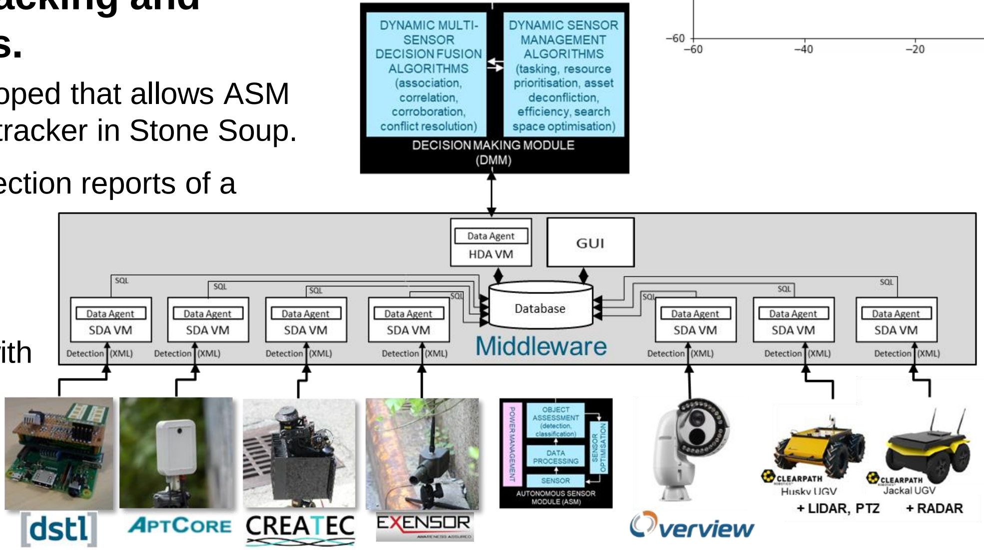
- Note: synthetic parameters used to avoid classification restrictions

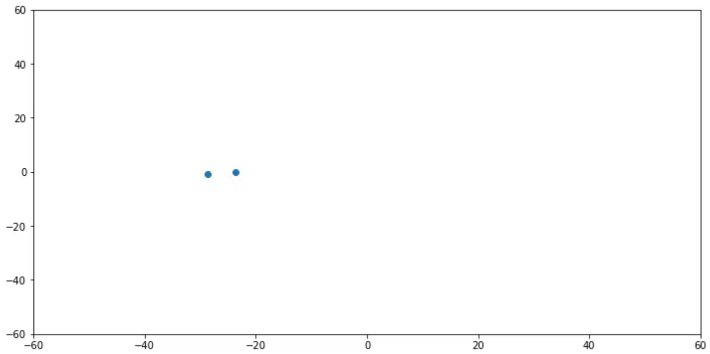
- **Example courtesy of Oliver Harrald (Dstl)**

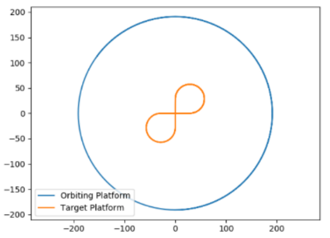


SAPIENT Interface

- **SAPIENT is a experimental multi-sensor fusion system using distributed sensors with smart local processing, connected to a fusion node that performs tracking and Situational Awareness.**
  - Stone Soup interface developed that allows ASM messages to be piped to a tracker in Stone Soup.
  - Basic tracker receiving Detection reports of a simulated target.
  - Target moves in figure of 8.
  - Sensor moves around it, sending detection reports with the target's state, detection time, confidence, etc.







- **Courtesy of Sebastian Vidal, Oliver Harrald and Steve Hiscocks (Dstl)**

Impact

FFI Forsvarets forskningsinstitutt Norwegian Defence Research Establishment

DRDC RDDC

U.S. NAVAL RESEARCH LABORATORY

Australian Government Department of Defence Defence Science and Technology Group

EURASIP European Association for Signal Processing

NATO OTAN

ISIF International Society of Information Fusion

The University Of Sheffield.

The Technical Cooperation Program TTCP

Fraunhofer

University of Liverpool

Ultra Electronics

STEWG

TELECOM SudParis

Heriot Watt University

LiU Linköping University

IP PARIS

TEXAS The University of Texas at Austin

L3HARRIS™

STR Systems & Technology Research

MBDA Missile Systems

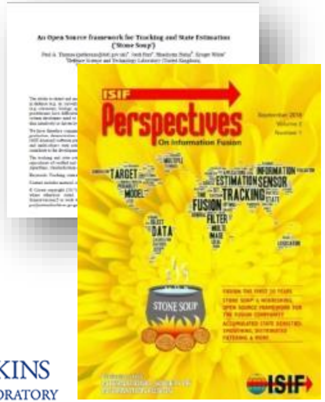
Carleton University

Leonardo

Airbus

KIT Karlsruher Institut für Technologie

APL Johns Hopkins Applied Physics Laboratory



What next?

[dstl] The Science Inside

Forthcoming additions (Pull Requests)

- PHD Evidential Filter
- Orbital state
- Rao-Blackwellised Particle Filter
- Tree-structures for gating
- IMM / GPB2
- Track Before Detect
- Run Manager

Future direction

- Multi-Hypothesis Tracker
  - Multi-Frame Assignment
- Posterior Cramer-Rao Lower Bound (PCRLB)
- Gaussian Processes
- Graphical User Interface (GUI)

Needs

- **Larger review community**
- **More papers based on Stone Soup**
- **Standard data sets**

Implications for the community

- ‘Snake oil’ filter for algorithm claims
- Rapid prototyping
- Accelerated Personal Development
- Algorithm ‘benchmarks’
- Repository of standard versions of algorithms
- Step towards a sharing culture
- Standard data sets
- Lowers industry’s perceived risk for algorithm adoption

However…

- Customers will demand higher performance

[dstl] The Science Inside

Discover more