BYU MAGICC LAB

C-UAS
CENTER FOR UNMANNED AIRCRAFT SYSTEMS

Vision Based Navigation and Tracking with Small UAS

Randal Beard

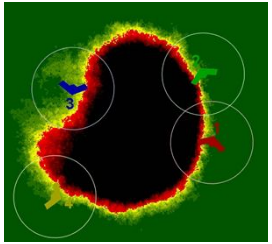
Brigham Young University

Outline

- Introduction and Motivation
- GPS Degraded Navigation
- Robust Target Tracking

UAS Research @ BYU

BYU MAGICC LAB



Cooperative Control

- Cooperative timing problems
- Cooperative persistent imaging
- Cooperative fire monitoring
- UAV/UGV Coordination
- Consensus seeking

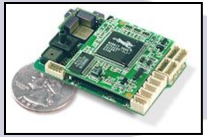
Path Planning
Trajectory Generation

- 3D Waypoint path planning
- Wind compensation
- Collision avoidance
  - Optic flow sensor
  - Laser ranger
  - EO cameras

Image/Sensor Directed Control

- Image Stabilization
- Geo-location
- Vision aided tracking and engagement

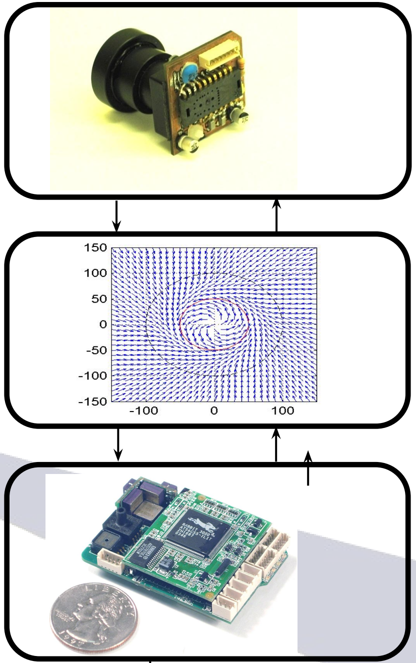
Autonomous Vehicles



- Autopilot design for small UAVs
- Attitude estimation
- Adaptive control
- Tailsitter guidance and control

C-UAS

Autonomous Flight For Miniature Air Vehicles



Sensor Based Flight

- Optic Flow
- Laser range finder
- EO/IR

Path Following

- Robust to wind
- Computationally efficient

Kestrel Autopilot

- Auto take-off, land
- Waypoint NAV
- GPS guided

Applications

- Canyon following
- Collision avoidance
- GPS denied navigation

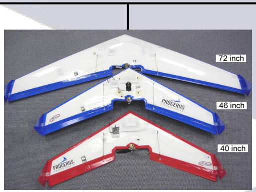
Applications

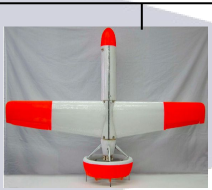
- Precision Navigation
- Target tracking
- Target geo-location

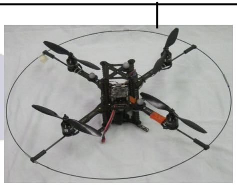
Technology Transition

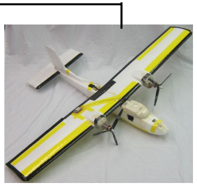
BYU MAGICC LAB ⇒ PROCERUS TECHNOLOGIES ⇒ LOCKHEED MARTIN Procerus Technologies, L.C.

2004 2012









18 Years of UAS Research at BYU

C-UAS

NSF

The National Science Foundation
Center for Unmanned Aircraft Systems

C-UAS

CU

Virginia Tech 1872

BRIGHAM YOUNG UNIVERSITY · FOUNDED BYU 1875 · PROVO, UTAH

Georgia Tech

MICHIGAN

Industry Members

facebook. INSITU 2d3 sensing BOEING AECOM

AFRL THE AIR FORCE RESEARCH LABORATORY LEAD | DISCOVER | DEVELOP | DELIVER

TEXTRON

U.S. DEPARTMENT OF THE INTERIOR BUREAU OF RECLAMATION

NAVAIR

ARL

NORTHROP GRUMMAN

United Technologies Research Center

UTOPIACOMPRESSION

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION NOAA U.S. DEPARTMENT OF COMMERCE

L3

NASA

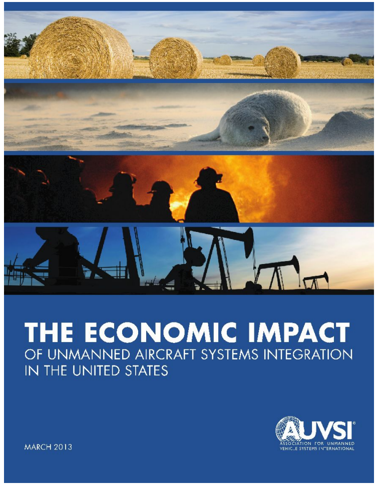
SINCLAIR COMMUNITY COLLEGE

AV aerovironment

bp

Unmanned Aircraft Impact

- Projected impact of UAS tremendous
  - Job creation – over 100,000 new jobs
  - Economic growth – $82B
- Unmanned aircraft are a game changer
  - Demonstrated convincingly in military applications
  - Widespread impact expected in commercial applications
  - Question: Not if, but how soon and to what degree?
- Numerous challenges to overcome – *opportunities for innovation*



THE ECONOMIC IMPACT
OF UNMANNED AIRCRAFT SYSTEMS INTEGRATION
IN THE UNITED STATES

MARCH 2013

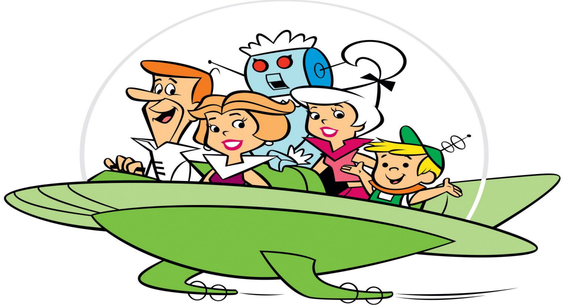
AUVSI
ASSOCIATION FOR UNMANNED VEHICLE SYSTEMS INTERNATIONAL

Envisioned Future for UAS/Autonomous Systems

BYU MAGICC LAB

Unmanned aircraft

Increasing safety, reliability, autonomy, public acceptance



Personal self-piloted aircraft

Self-driving cars

C-UAS

Significant Technical Gaps

Safe and reliable autonomous flight

- Robustness to loss/degradation of GPS

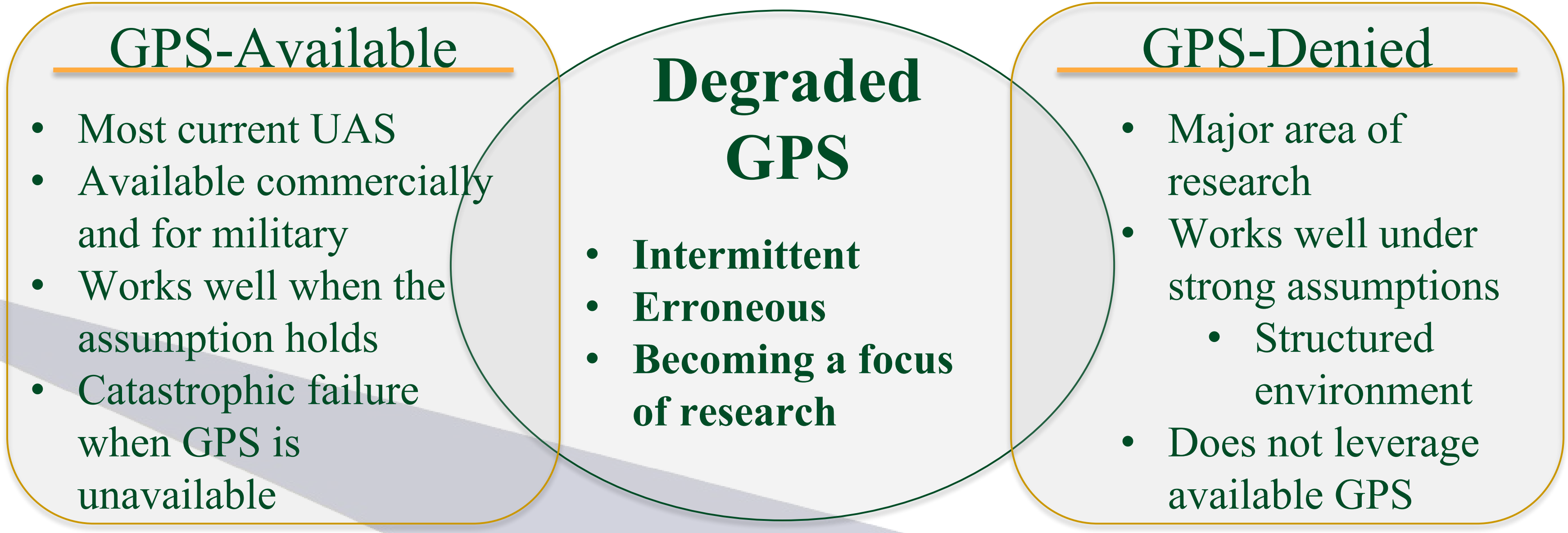
Increased capabilities in range, payload, agility

- Robust tracking of ground based targets

Outline

- ➢ Introduction and Motivation
- ➢ GPS Degraded Navigation
- ➢ Robust Target Tracking

Motivation & Background



GPS-Available

- Most current UAS
- Available commercially and for military
- Works well when the assumption holds
- Catastrophic failure when GPS is unavailable

Degraded GPS

- **Intermittent**
- **Erroneous**
- **Becoming a focus of research**

GPS-Denied

- Major area of research
- Works well under strong assumptions
  - Structured environment
- Does not leverage available GPS

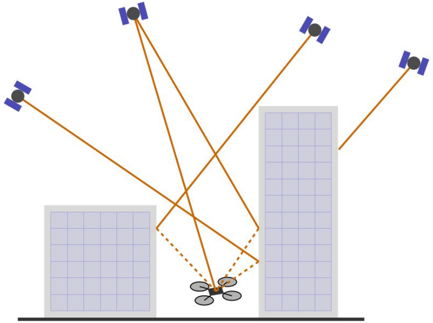
GPS-degradation

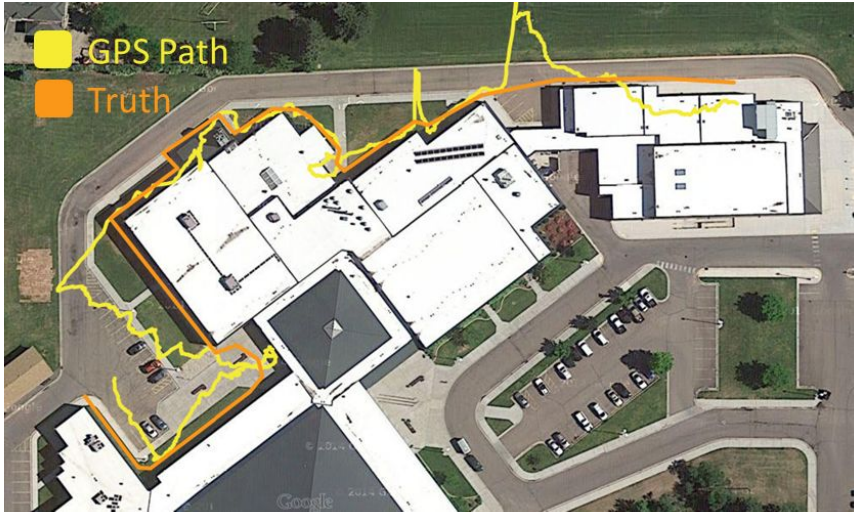
Availability:

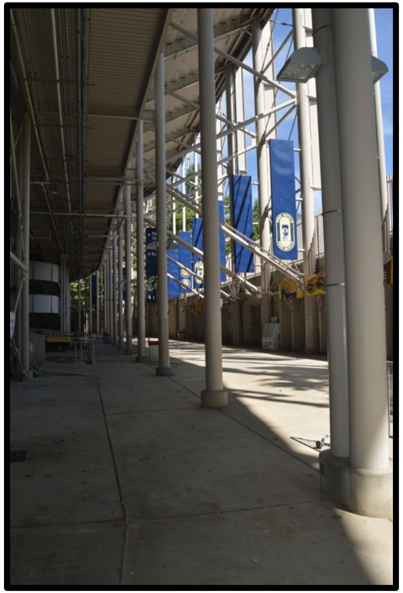
- Indoors
- Urban canyon
- Sensor failure
- Foliage
- Jamming

Uncertainty:

- Atmospheric delays
- Dilution of precision
- Number of satellites
- Multipath
- Spoofing





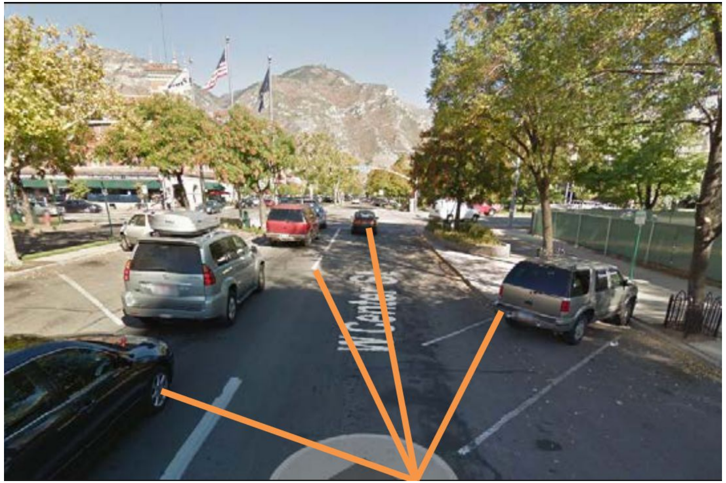


Objective:

Allow UAS to operate robustly in any environment with intermittent or degraded GPS.

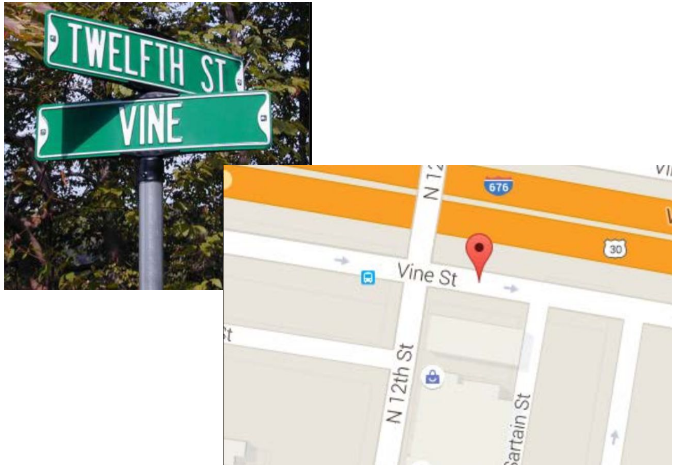
Relative navigation intuition

Driver



Estimate state and plan actions with respect to local environment.

Passenger “Riding Shotgun”



Observe landmarks (signs), reference map, plan high-level route.

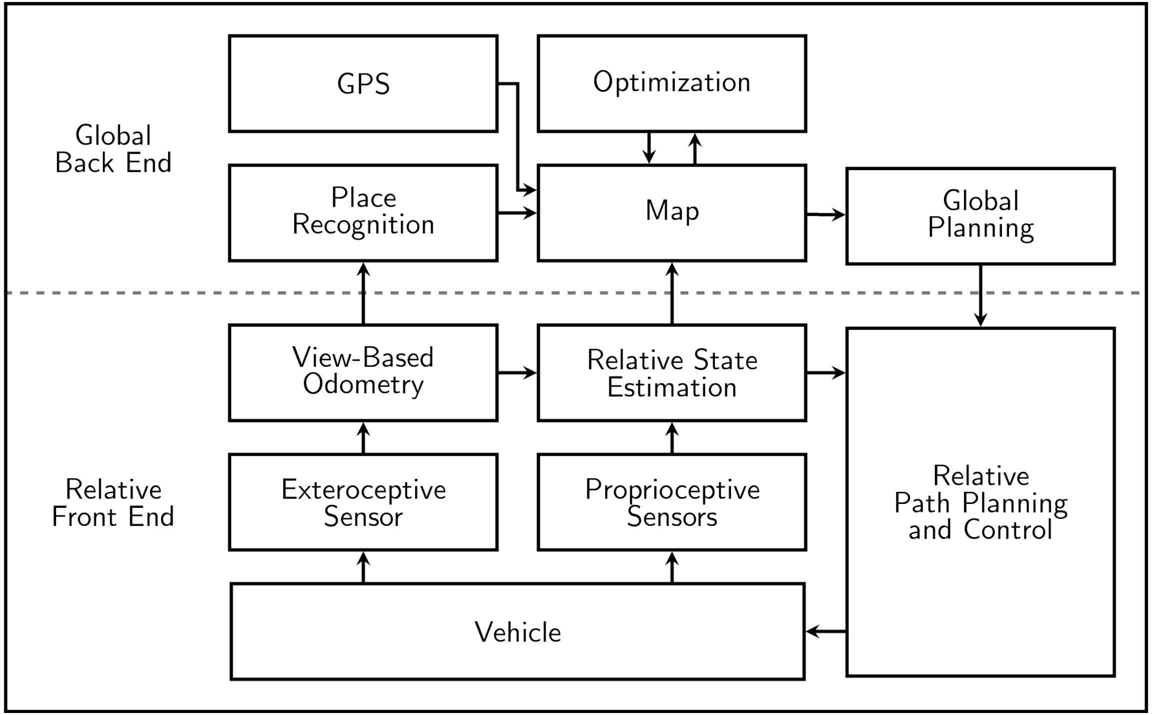
Low frequency, relative directions

Relative navigation architecture

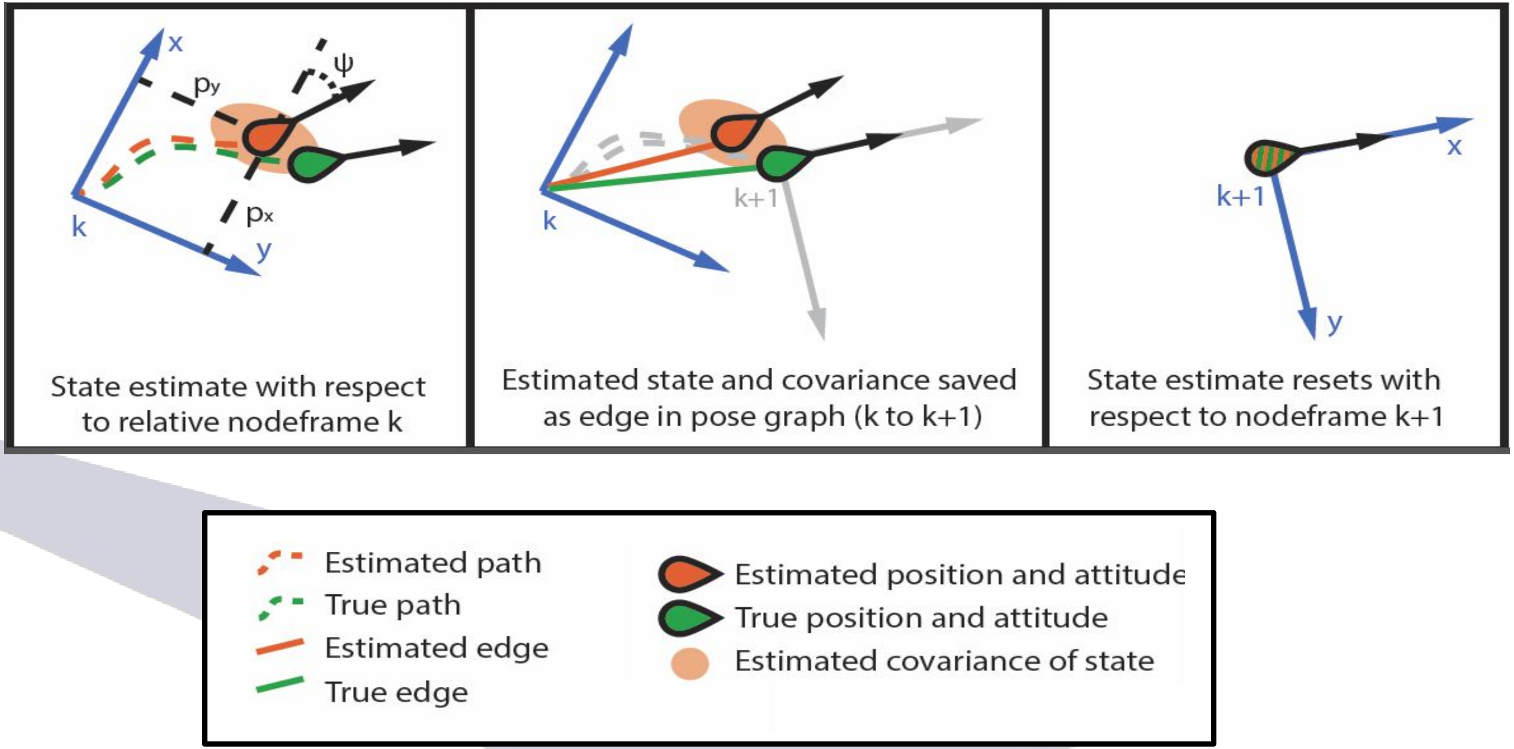
Flight critical front-end decoupled from backend (i.e. observable, real-time, onboard)

Not specify to any one platform, sensor suite, estimation and control strategy.

Back-end scalable for distributed multi-agent cooperation.



Relative Navigation Framework


x
p_y
ψ
k
p_x
y
k
k+1
k+1
x
y

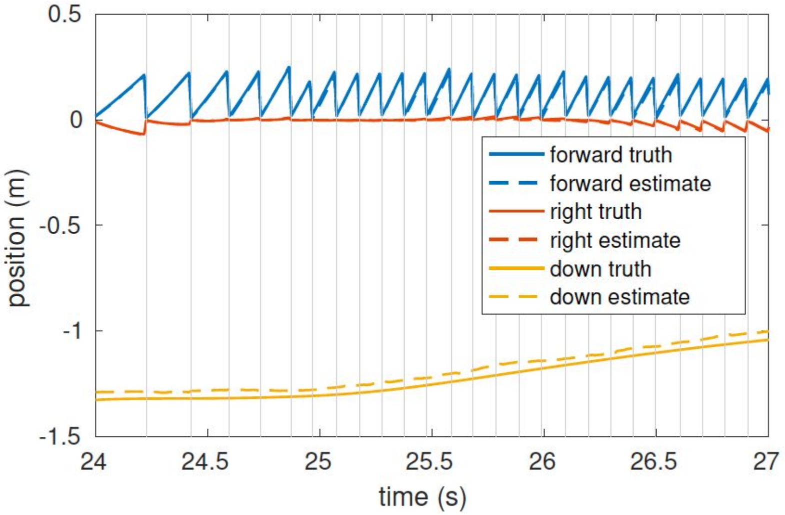
State estimate with respect to relative nodeframe k

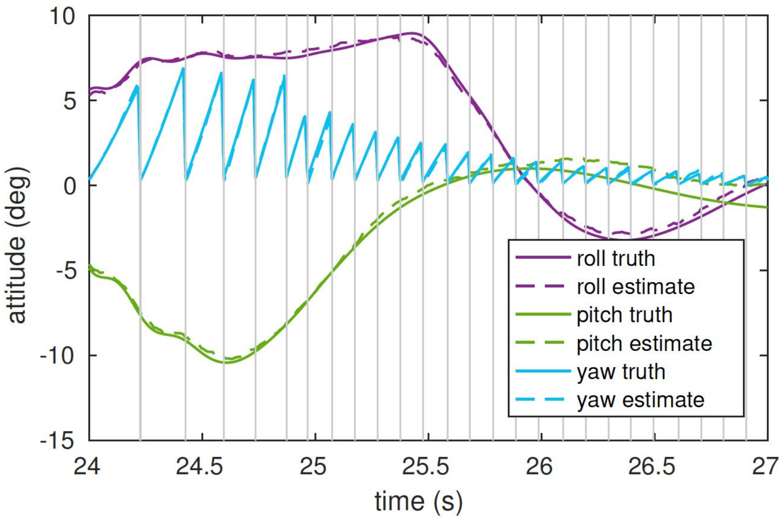
Estimated state and covariance saved as edge in pose graph (k to k+1)

State estimate resets with respect to nodeframe k+1

Estimated path
True path
Estimated edge
True edge
Estimated position and attitude
True position and attitude
Estimated covariance of state

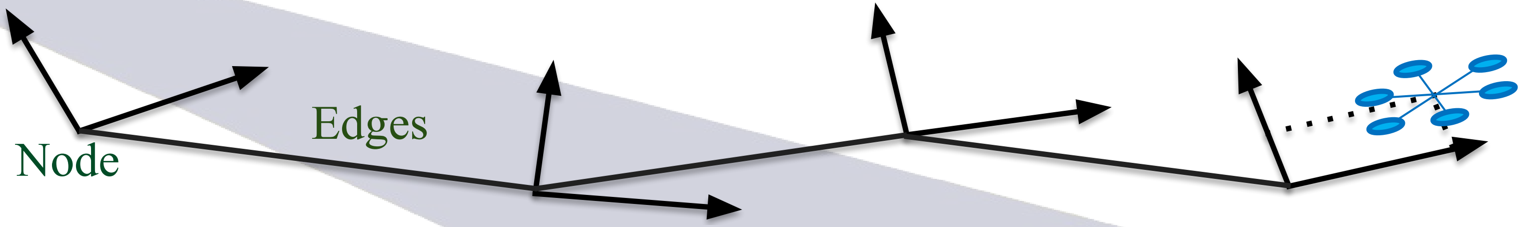
Example State Estimates





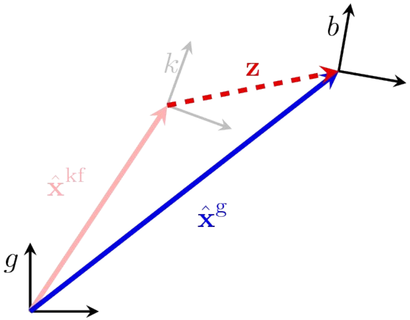
Relative Navigation Framework

- Back End
  - Constructs graph from front-end using odometry
  - OpenFABMAP for place recognition between nodeframes
    - or GPS measurement
  - g2o for graph optimization



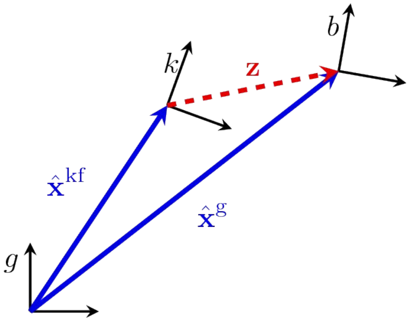
Approach comparison

BYU MAGICC LAB



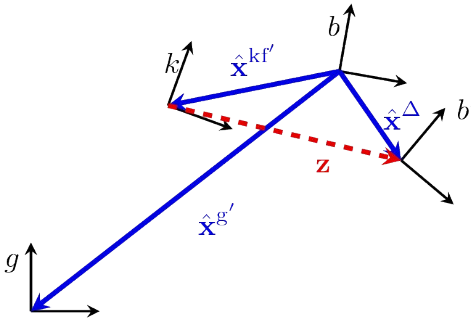
b
k
$\mathbf{z}$
$\hat{\mathbf{x}}^{\mathrm{kf}}$
$\hat{\mathbf{x}}^{\mathrm{g}}$
g

Pseudo-global



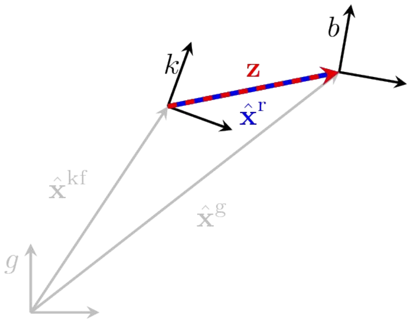
b
k
$\mathbf{z}$
$\hat{\mathbf{x}}^{\mathrm{kf}}$
$\hat{\mathbf{x}}^{\mathrm{g}}$
g

Stochastic cloning



b
k
$\hat{\mathbf{x}}^{\mathrm{kf}'}$
$\hat{\mathbf{x}}^{\Delta}$
b'
$\mathbf{z}$
$\hat{\mathbf{x}}^{\mathrm{g}'}$
g

Robocentric

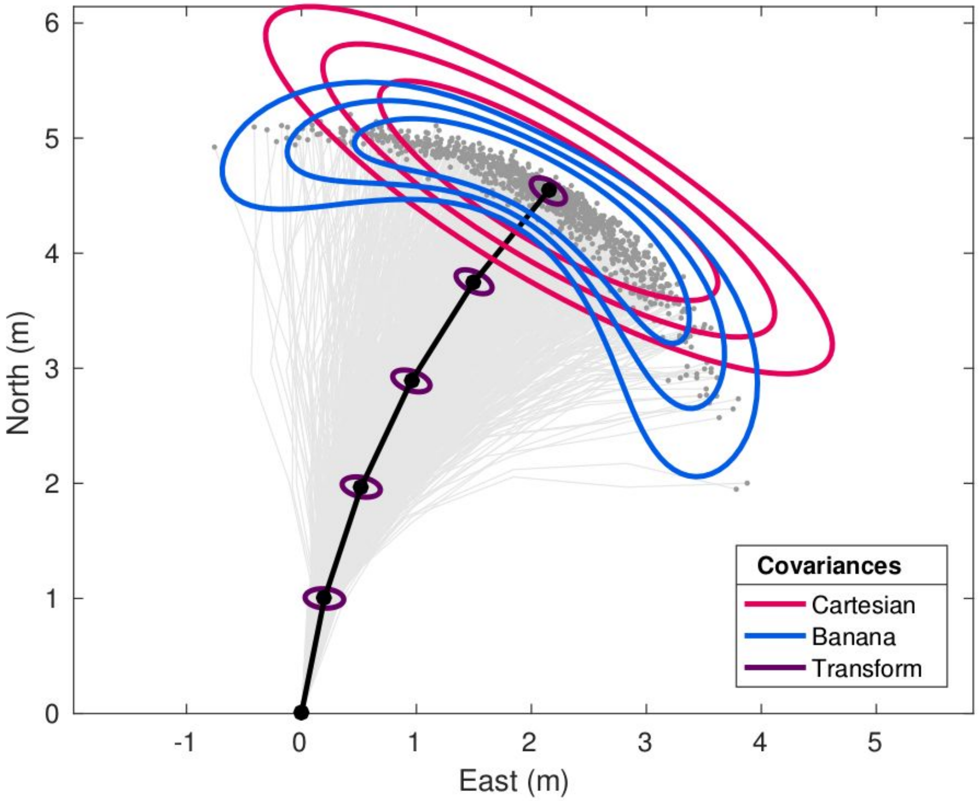


b
k
$\mathbf{z}$
$\hat{\mathbf{x}}^{\mathrm{r}}$
$\hat{\mathbf{x}}^{\mathrm{kf}}$
$\hat{\mathbf{x}}^{\mathrm{g}}$
g

Relative Navigation

C-UAS
CENTER FOR UNMANNED AIRCRAFT SYSTEMS

Global Reconstruction



Sample Covariance:

$$\Sigma = \frac{1}{N}\sum_{n}^{N} \mathbf{e}_n \mathbf{e}_n^{\mathsf{T}}$$

Cartesian coordinates (ellipse)

$$\begin{aligned} \mathbf{e} &= \ominus \hat{\mathbf{x}} \oplus \mathbf{x} \\ &= \mathbf{x} - \hat{\mathbf{x}} \end{aligned}$$

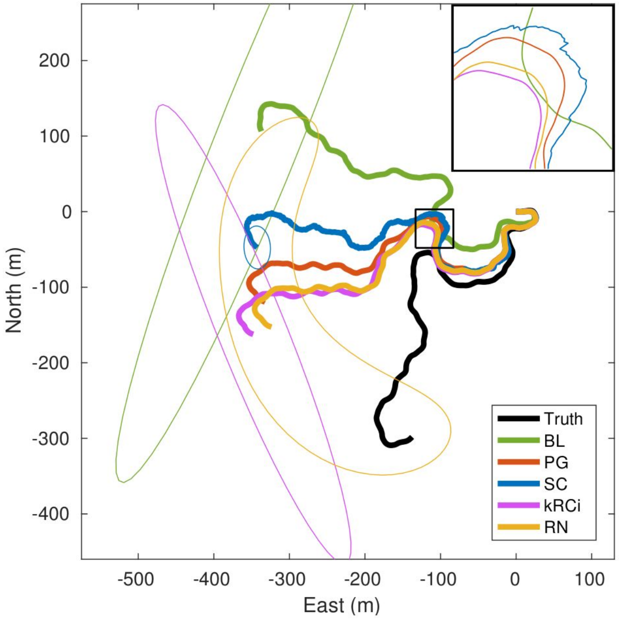
Exponential coordinates (banana)

$$\mathbf{e}_{\mathrm{RN}} = \log\left(\ominus \hat{\mathbf{x}} \oplus \mathbf{x}\right)$$

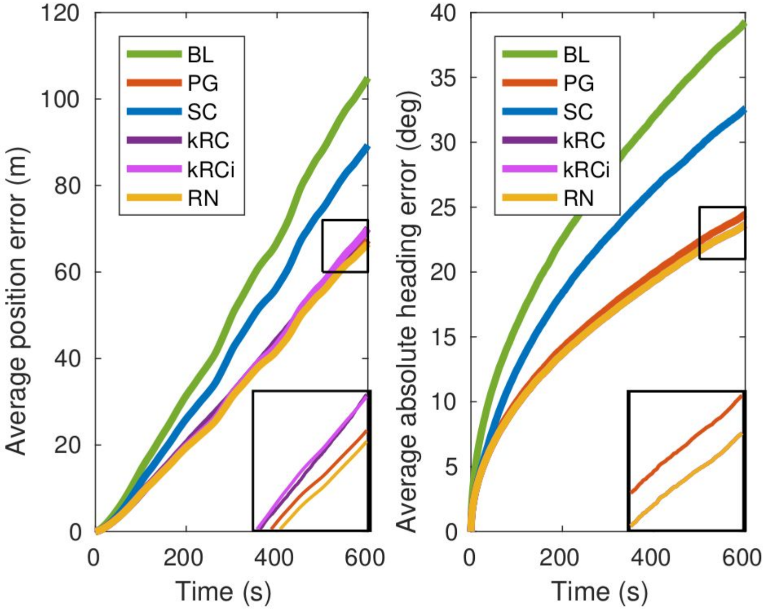
$$\log : SE(2) \to \mathfrak{se}(2)$$

Simulation performance

Example trajectory

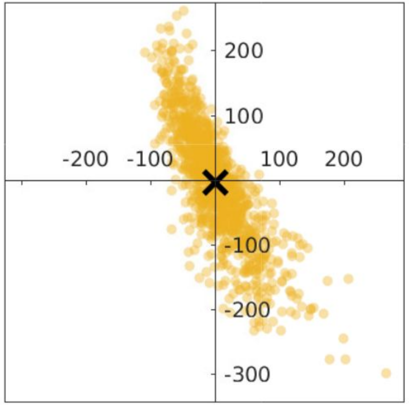


Accuracy



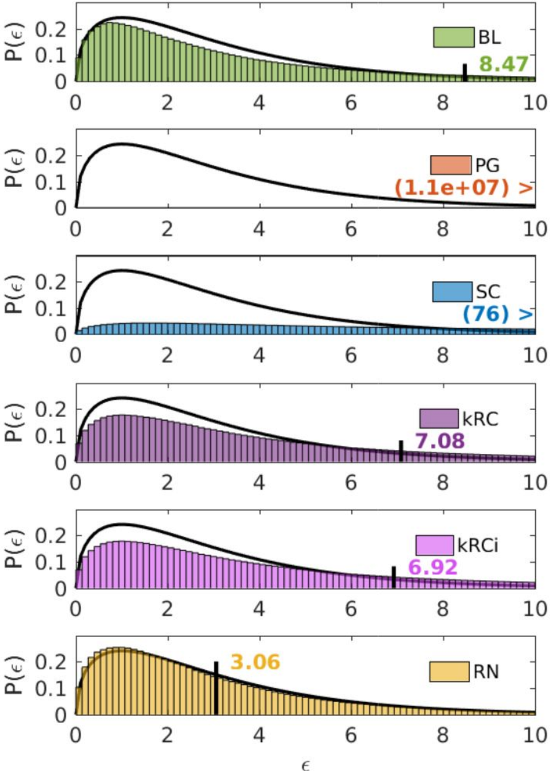
Simulation performance

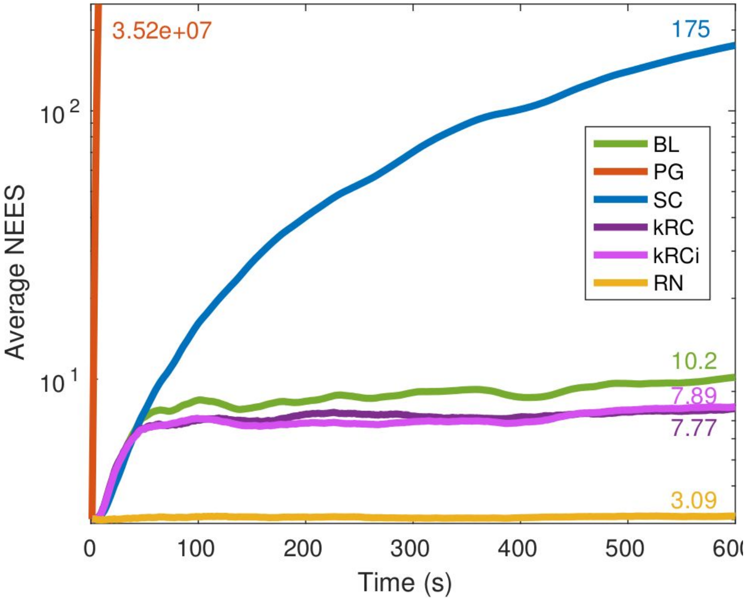
Consistency: Bias



(f) $\mathbf{e}_{\mathrm{RN}}$

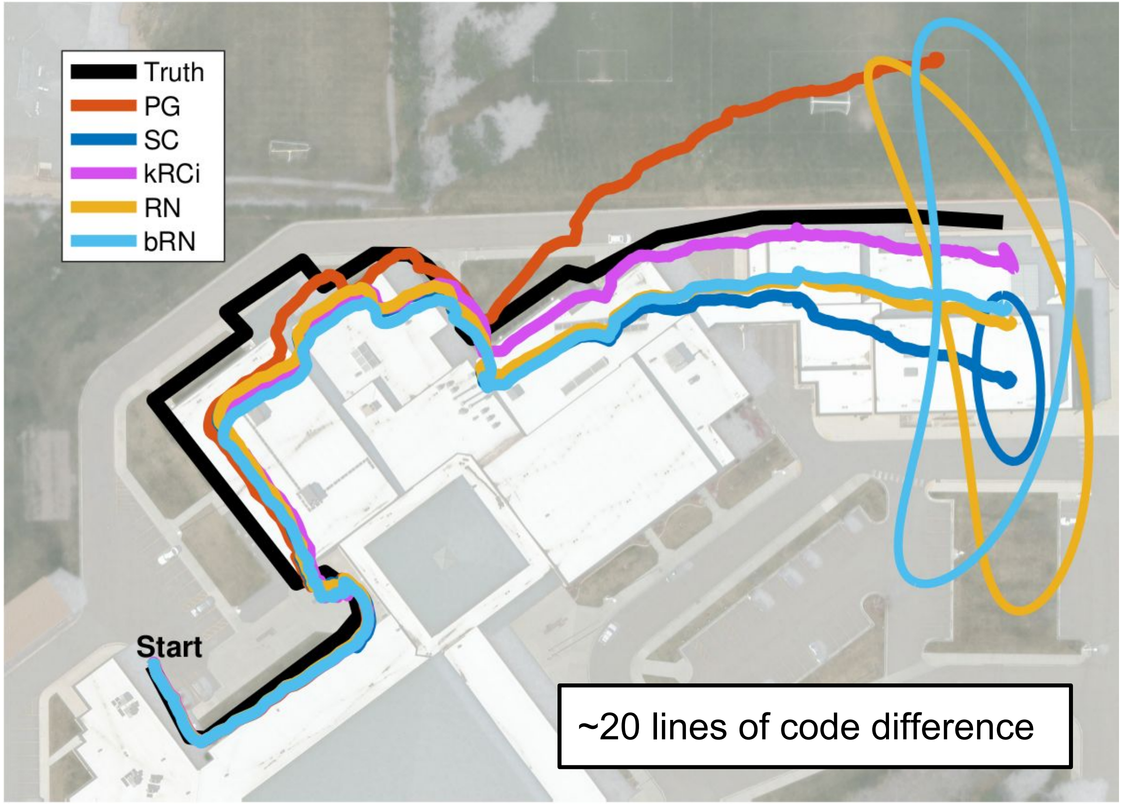
Consistency: Uncertainty





$$\epsilon_n = \left(\mathbf{x} - \hat{\mathbf{x}}_n\right)^{\mathsf{T}} \mathbf{P}^{-1} \left(\mathbf{x} - \hat{\mathbf{x}}_n\right)$$

Hardware performance



~20 lines of code difference

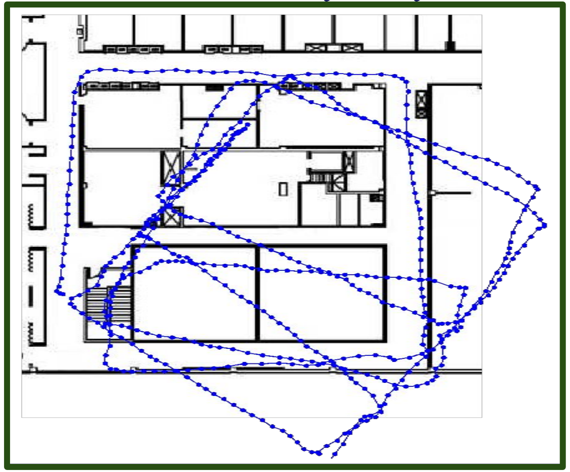
Indoor Flight Video

- ➢ 330 m, 12-minute flight
- ➢ Figure 8 pattern, 5 loops
- ➢ Components of architecture functioning as intended

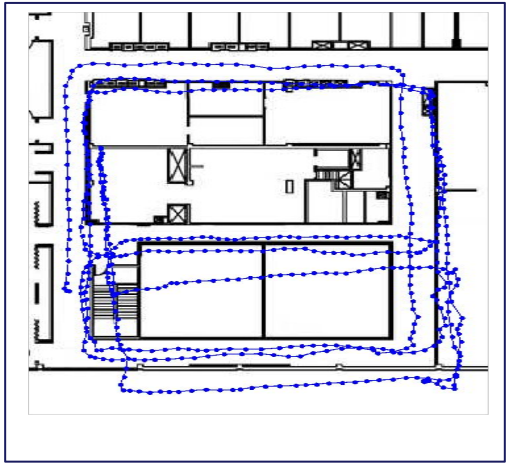
- Demonstrates performance of real-time relative front end
- Using visual odometry, IMU data to estimate states

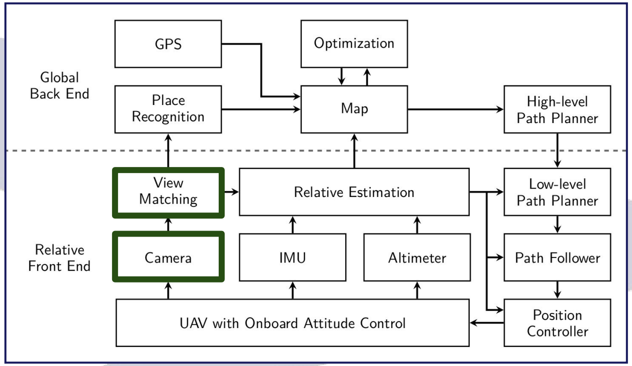
Relative Nav Front-End Results

Visual Odometry Only



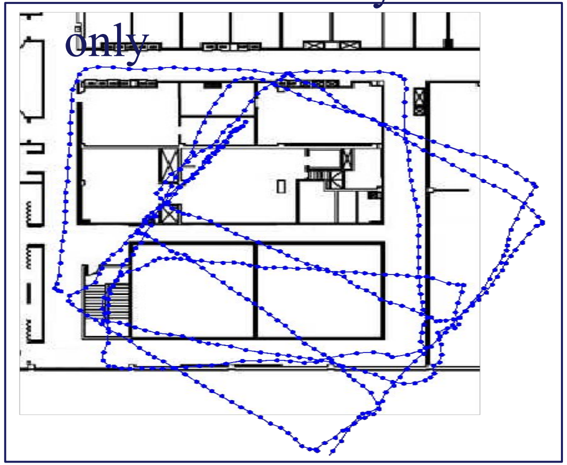
VO + MEKF



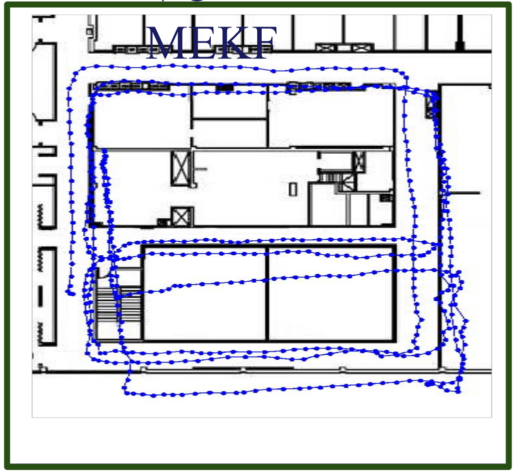


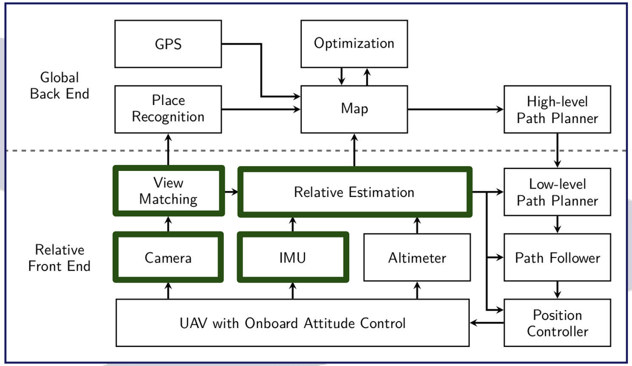
Relative Nav Front-End Results

Visual Odometry only



VO + MEKF





Global Back End

GPS

Optimization

Place Recognition

Map

High-level Path Planner

Relative Front End

View Matching

Relative Estimation

Low-level Path Planner

Camera

IMU

Altimeter

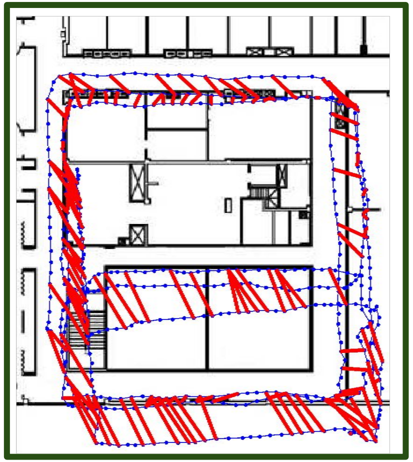
Path Follower

UAV with Onboard Attitude Control

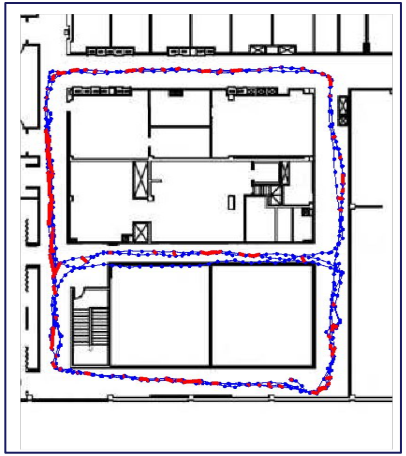
Position Controller

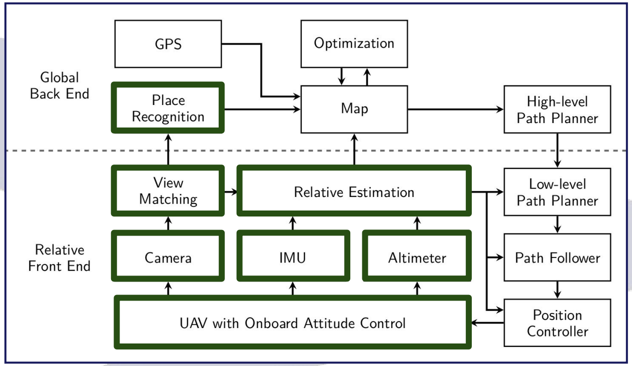
Relative Nav Back-End Results

Before Optimization



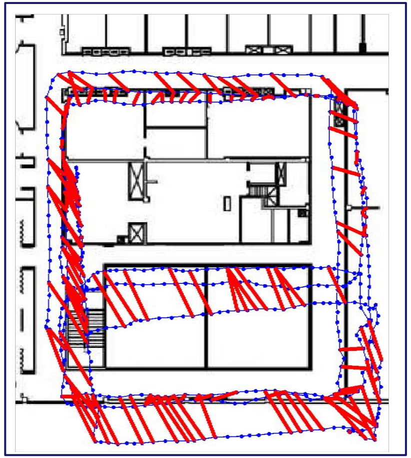
After Optimization



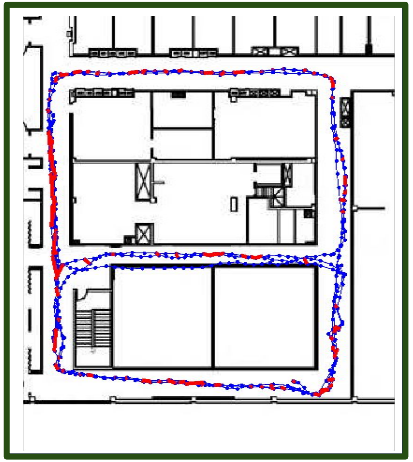


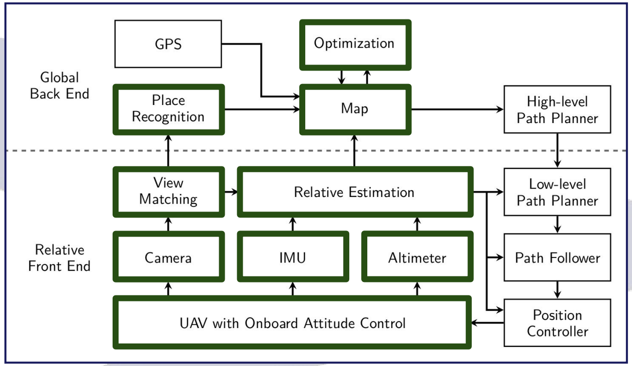
Relative Nav Back-End Results

Before Optimization



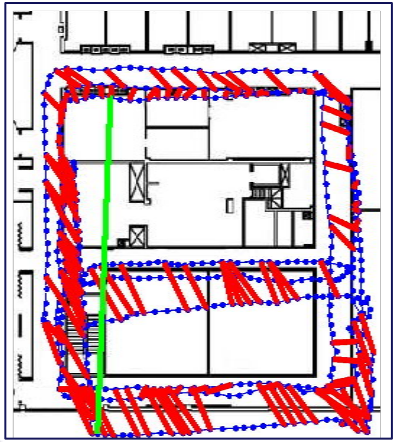
After Optimization



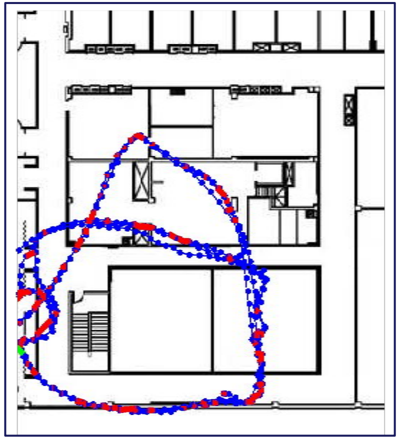


Back-End Improvements

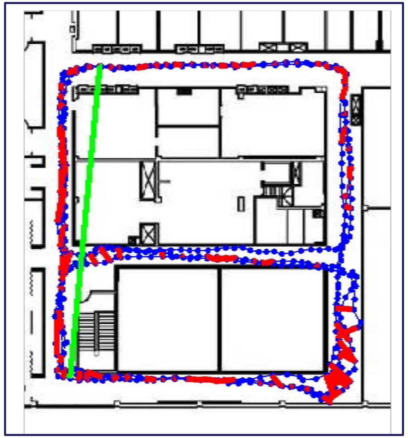
False-positive loop closure

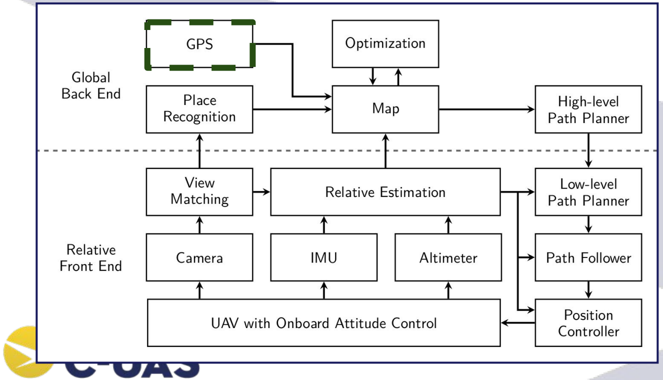


Standard LS optimization



Robust LS optimization

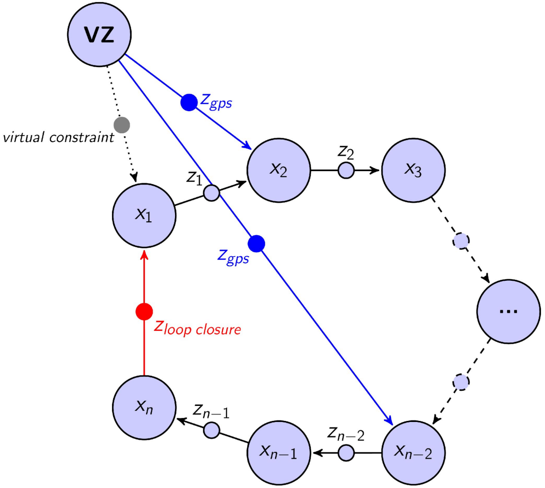


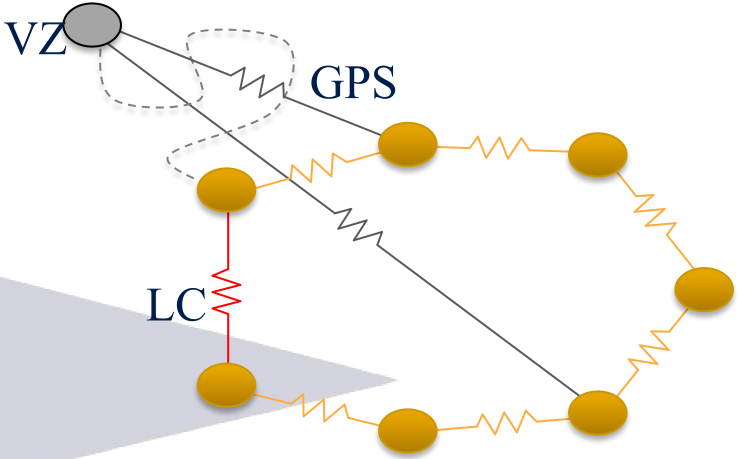


- Brittle to false-positive loop closures
- GPS not utilized
  - Must be robust to degraded or intermittent GPS

Incorporating GPS

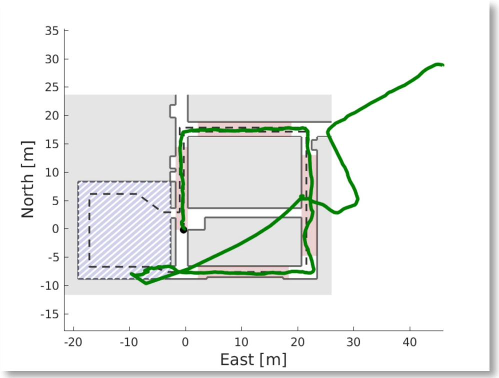
- Add a “virtual zero” node to the graph
  - Virtual zero represents the origin of an absolute global coordinate frame
- Constrain the virtual zero to another graph node with no certainty
  - This constraint represents the absolute global position of the corresponding node
  - Can be constrained to any node in the graph, we typically use the first node
- When a GPS measurement is received, add a new constraint between the receiving node and the virtual zero





Multi-Sensor Estimation

Single Sensor Estimation

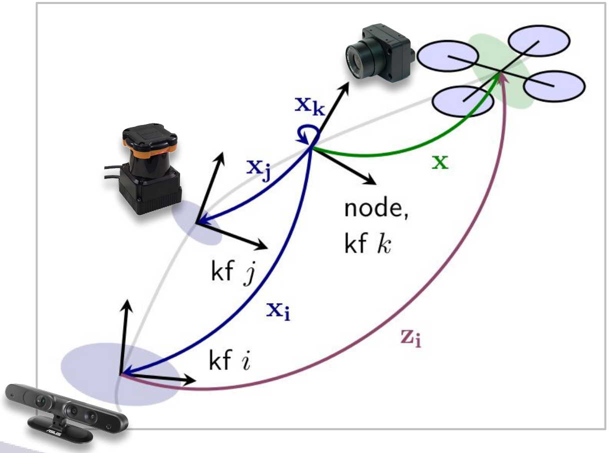


- Current state of the art
- Poor robustness
- Brittle to changes in environment
  - Lighting, structure, etc.

Multi-Sensor Estimation

- Complementary sensing modalities
- Redundancy
- Increased robustness

Multi-Sensor Estimation



Challenges

- Each sensor maintains its own keyframe at a given moment
- Measurement with respect to different coordinate systems

Solution

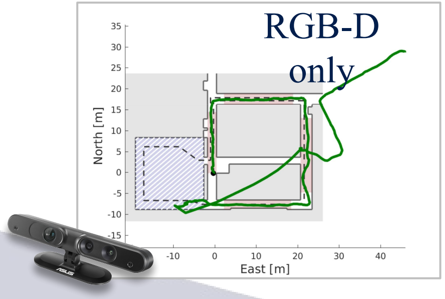
- Augment state with information about odometry keyframes
- EKF framework

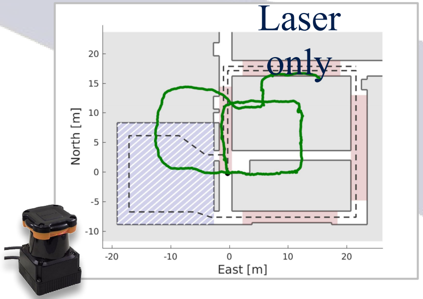
Advantages

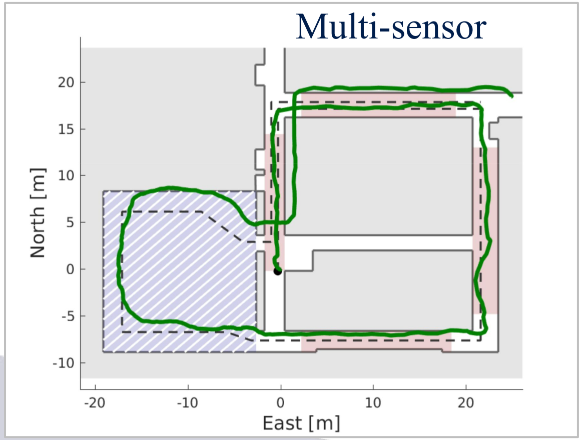
- Generic: can accept any odometry-like update
- Consistent: accounts for propagation of uncertainty through the system
- Allows for asynchronous keyframes between sensors
- Seamlessly handles odometry failure and re-initialization

Multi-Sensor Estimation

- Indoor environment, post-processed data
- Odometry sources:
  - Visual odometry with RGB-D camera (fails in dark room)
  - Scan matching with laser scanner (fails in long hallways)
- Manual gating of odometry in indicated regions

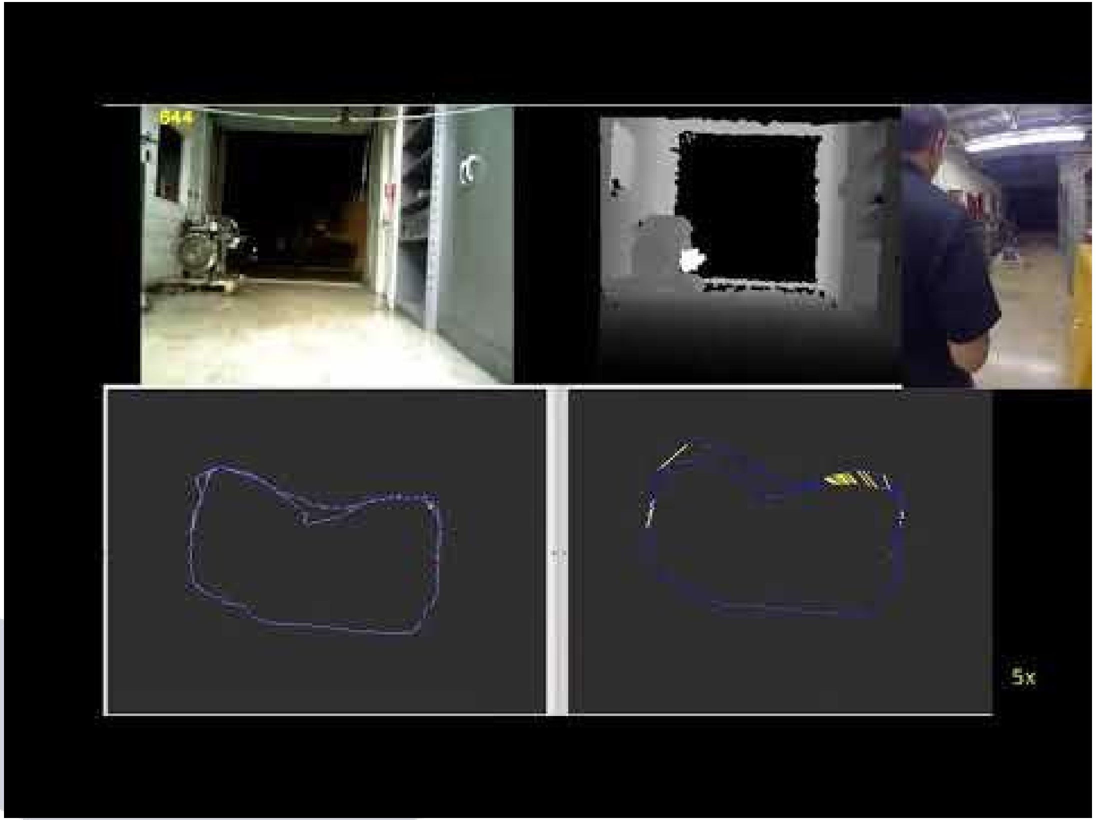






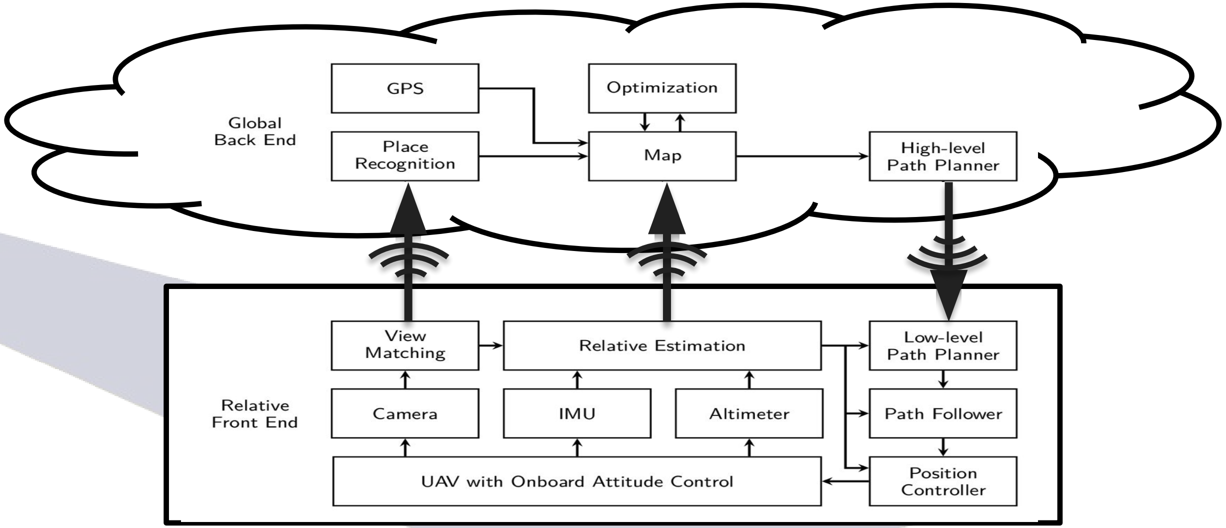
Flight Results

- 10+ minute flight.
- ~1 m error on first loop closure.
- Create globally-consistent metric map during flight.
- Maintains stability through optimization.
- Autonomously navigate through previously observed areas, including indoor/outdoor transition.



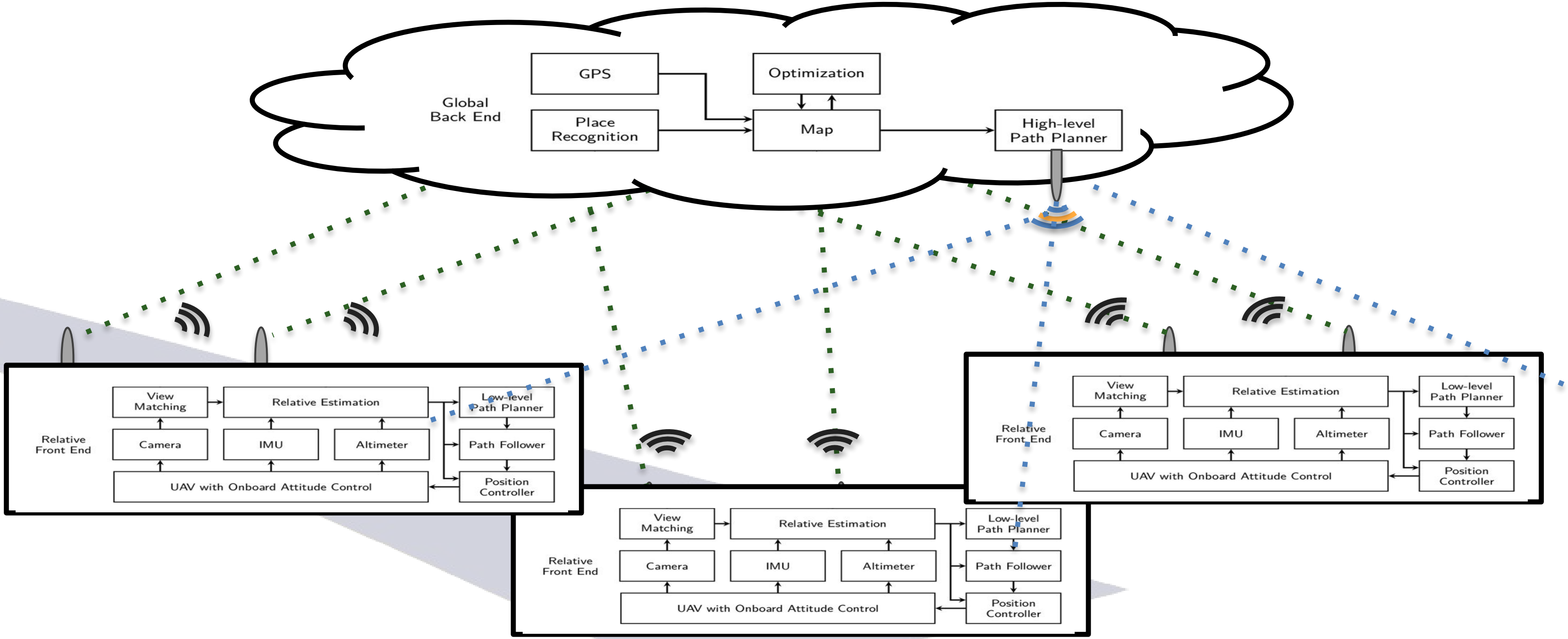
Cloud-Enabled Multi-Agent Operation

The Backend can be completely decoupled from the real-time estimation and control



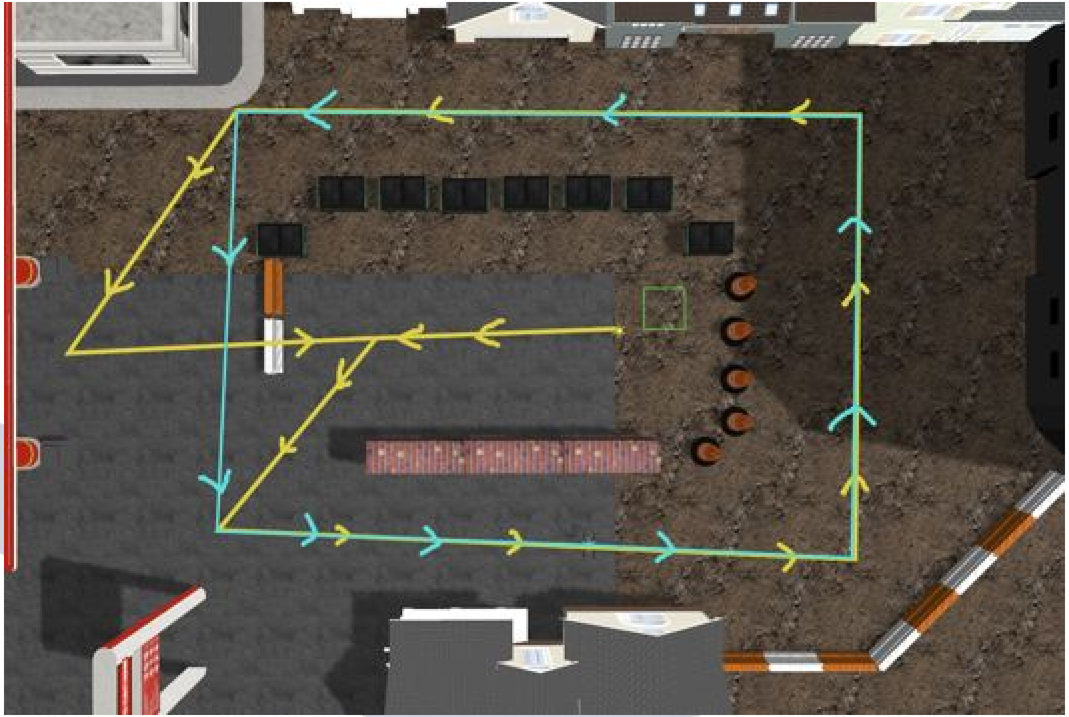
Cloud-Enabled Multi-Agent Operation

Use "cloud" resources to share a common backend

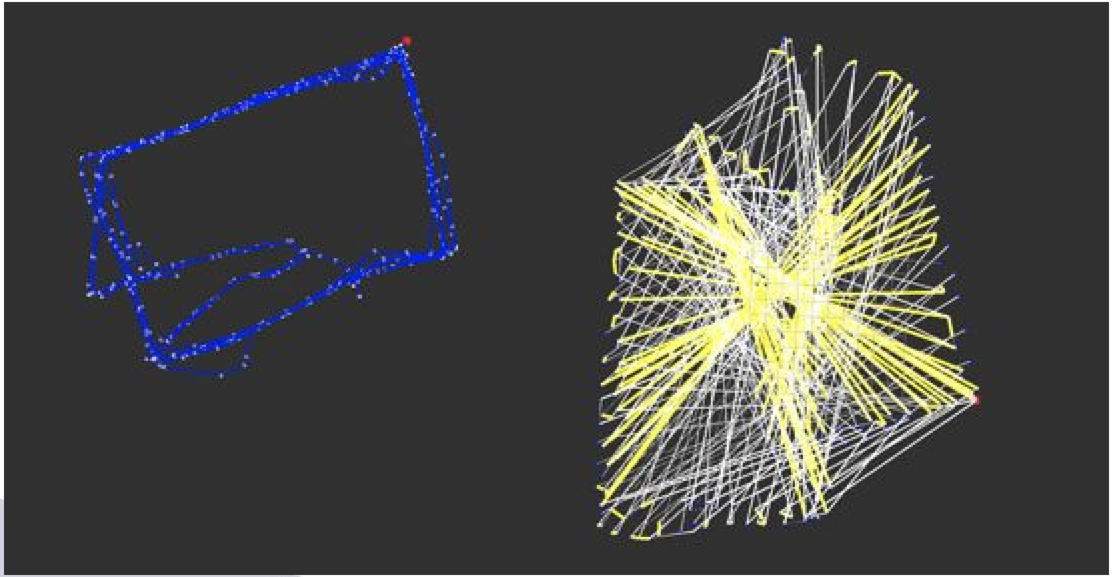


Multi-Agent Simulation

Two multirotors simulated in ROS/Gazebo. Sharing a backend in the cloud.



Multi-Agent Simulation



- Both multirotors were initialized with initial estimate of (0,0,0).
- Loop closures allowed global backend to solve for the optimized map containing odometry from both multirotors.
- No control jumps from state optimization

Implications

- Relative Navigation means map “slides around” underneath the agent, without disrupting its state estimation and control.

- Multiple agents can work together, communicating only through the cloud, and create optimized maps in real time.

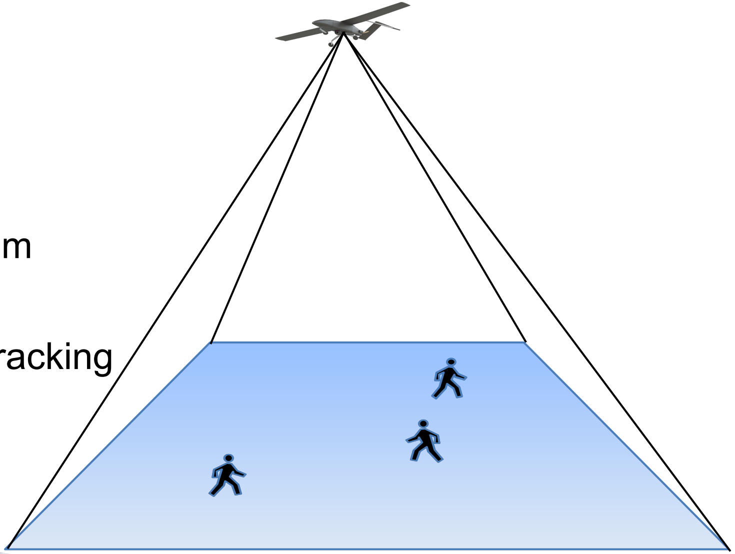
- State jumps from optimization do not result in control jumps or estimation jumps.

Outline

- ➢ Introduction and Motivation
- ➢ GPS Degraded Navigation
- ➢ Robust Target Tracking

Project Summary

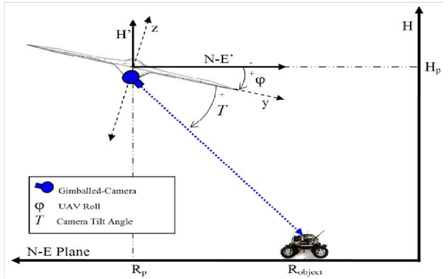
Main Objective: Increase the accuracy and robustness of visual tracking from UAS



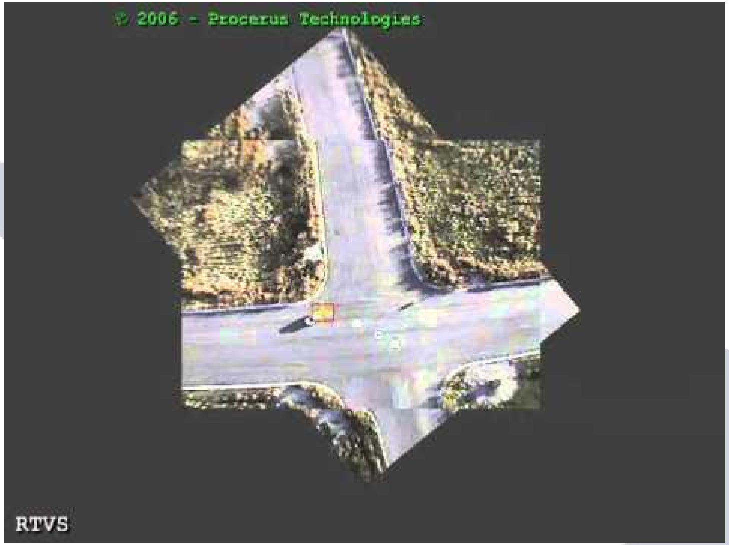
Technical Approach:

- Multiple Target Tracking - Recursive RANSAC Algorithm
- Computer vision techniques
- Flight control and gimballing algorithms that enhance tracking
- Human Interaction
- Exploit embedded GPUs for on-board processing
- Flight demonstrations

Motivation: Geolocation

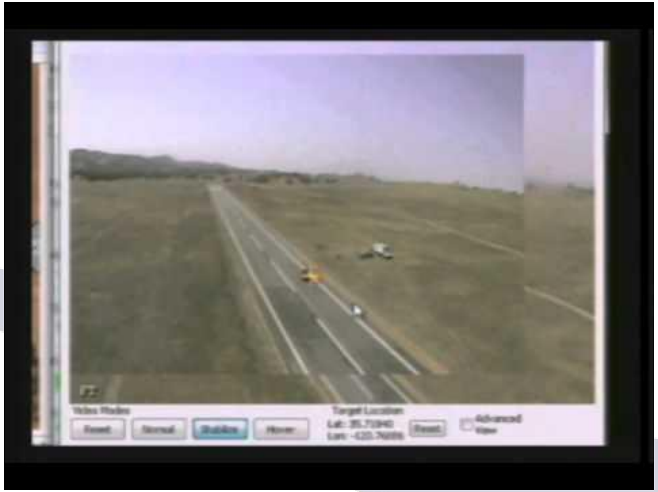


- Human identifies target in the image
- Use GPS, IMU, terrain model to geolocate target
- Maneuver UAS and gimbal to keep the target in the camera field of view.



- Note that video tracker gets stuck on manhole.
- Human intervention is required to re-acquire the truck.
- Frequent interaction required.

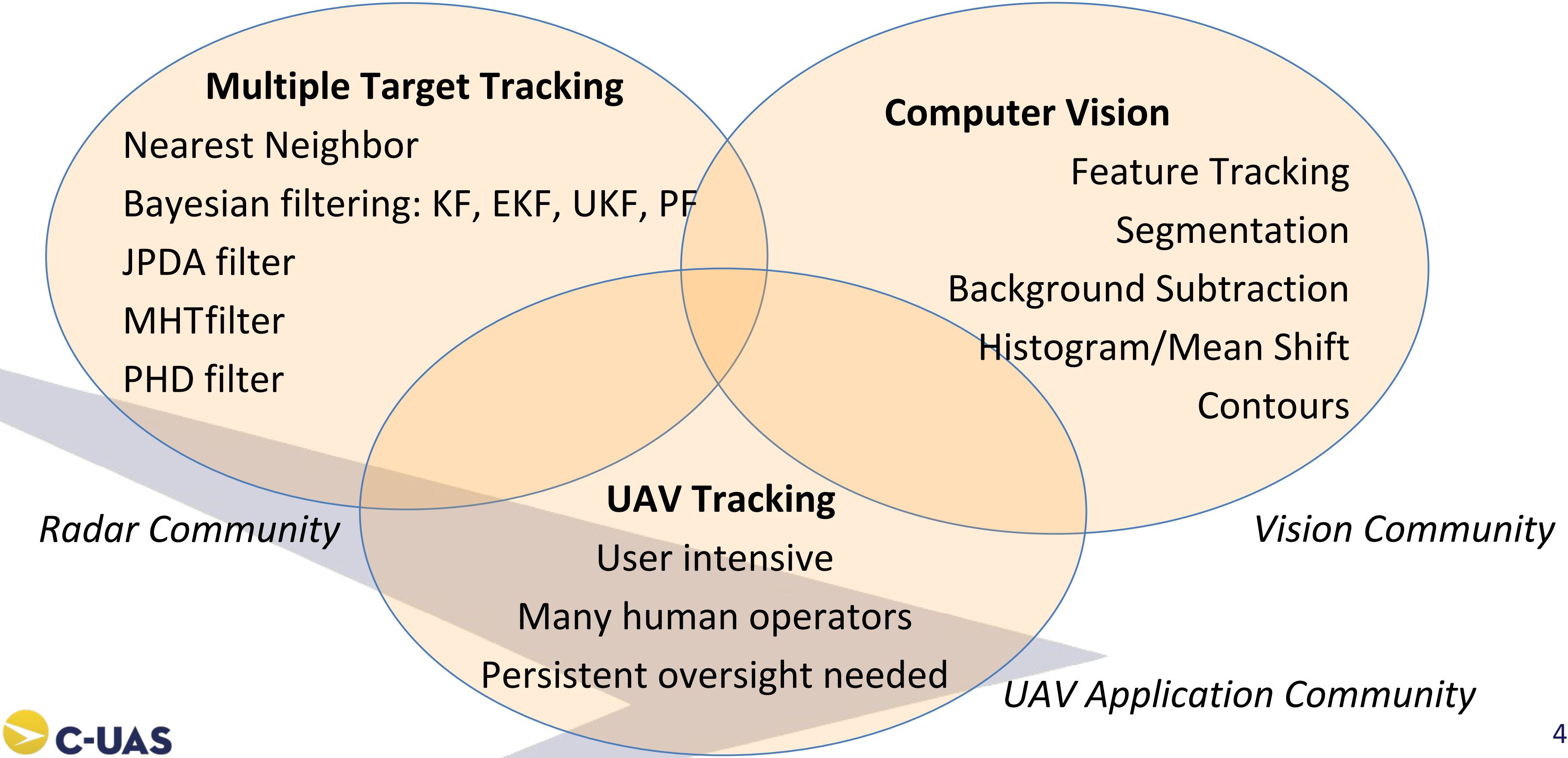
Motivation: Precision Landing



Target Location
Lat: 35.71940
Lon: -120.76006

- User guides the vehicle through video interface
- Off-board vision procession
- Frequent human interaction is required because the tracker is not robust.

Current State of the Art



Multiple Target Tracking

Nearest Neighbor

Bayesian filtering: KF, EKF, UKF, PF

JPDA filter

MHTfilter

PHD filter

Computer Vision

Feature Tracking

Segmentation

Background Subtraction

Histogram/Mean Shift

Contours

UAV Tracking

User intensive

Many human operators

Persistent oversight needed

Radar Community

Vision Community

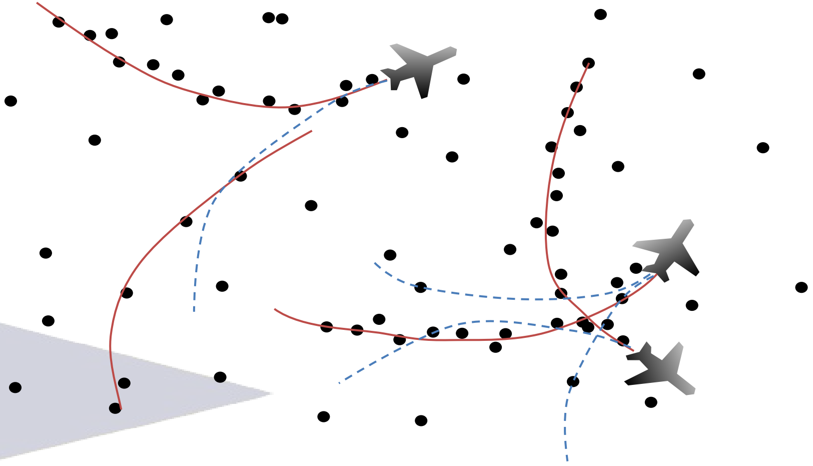
UAV Application Community

Outline

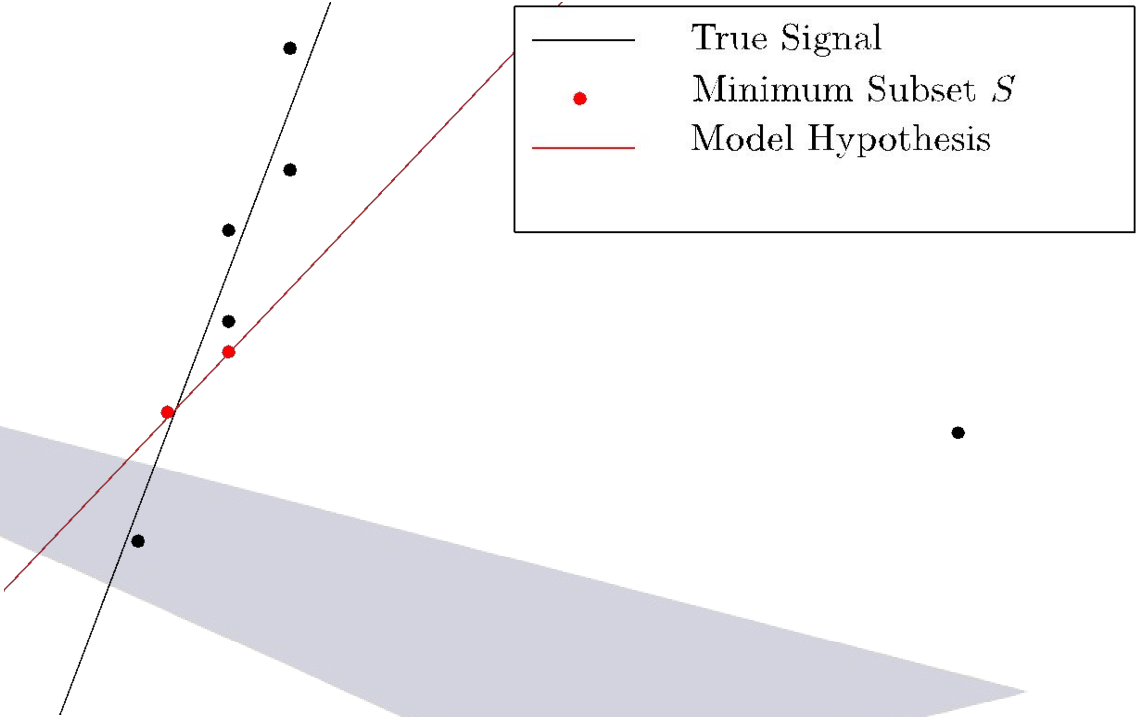
- ➢ Introduction and Motivation
- ➢ GPS Degraded Navigation
- ➢ Robust Target Tracking
  - – Recursive RANSAC
  - – Track Continuity
  - – Inertial vs image tracking - SLAM
  - – Flight control and gimbal servoing
  - – Safe2Ditch
  - – Hardware implementation

Recursive-RANSAC

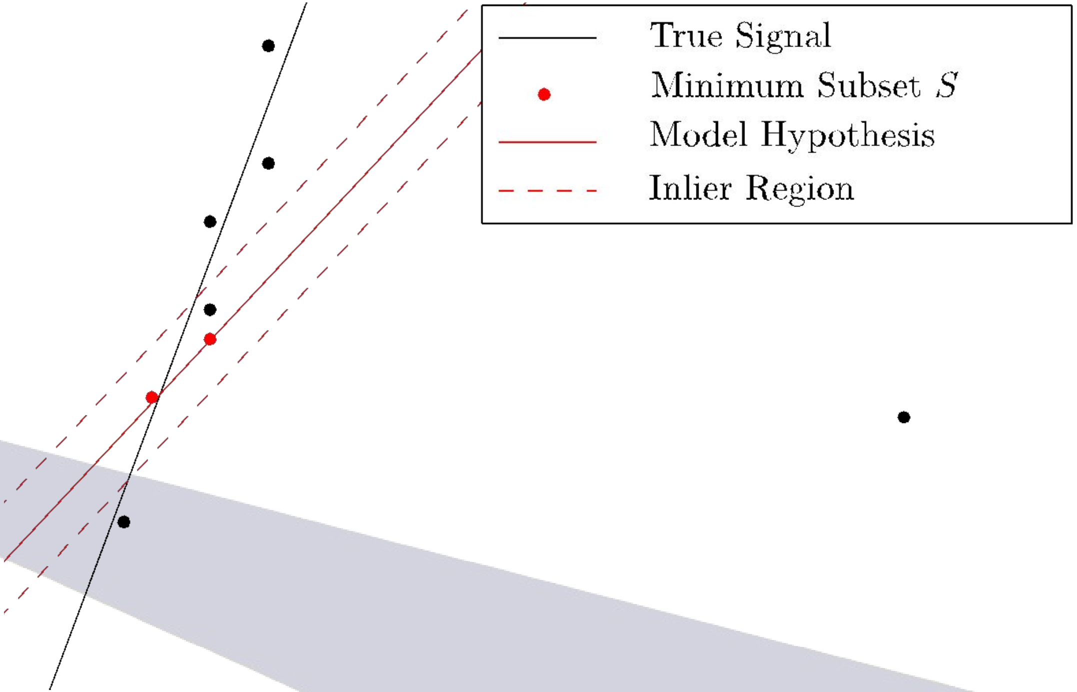
- New algorithm – developed at BYU
- Uses fast random search to find potential tracks
- Produces many potential tracks; keeps those most consistent with data
- Fast data association mechanism
- Excellent track continuity
- Robust performance in cluttered data



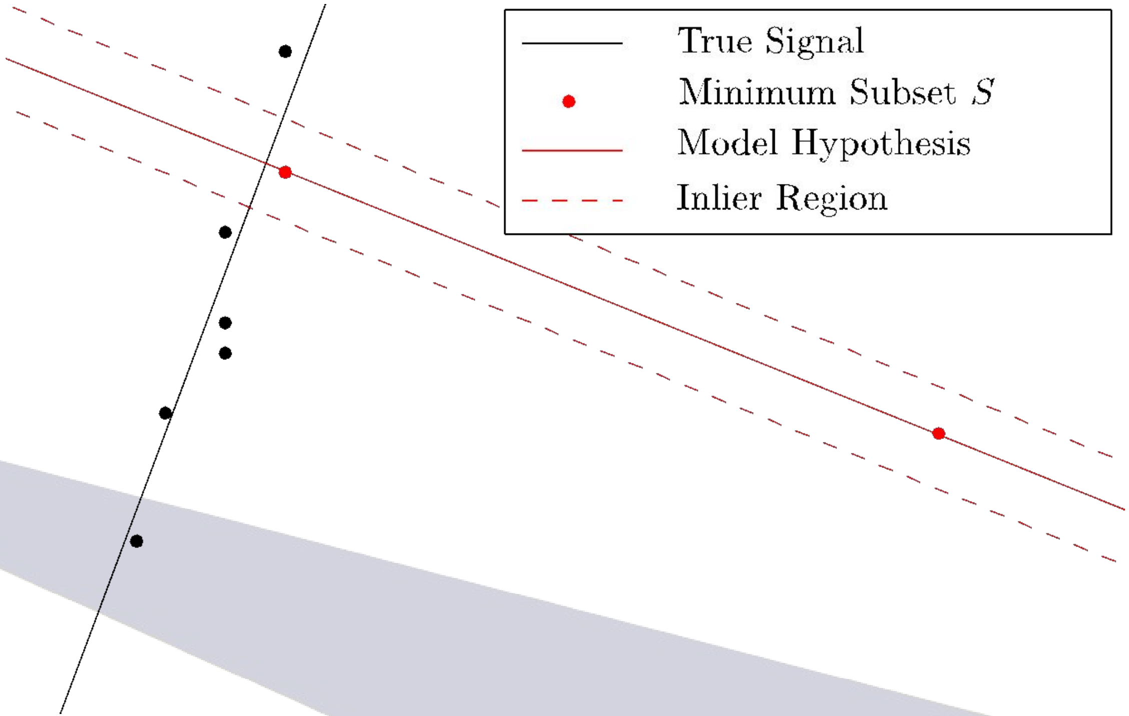
Random Sample Consensus (RANSAC)



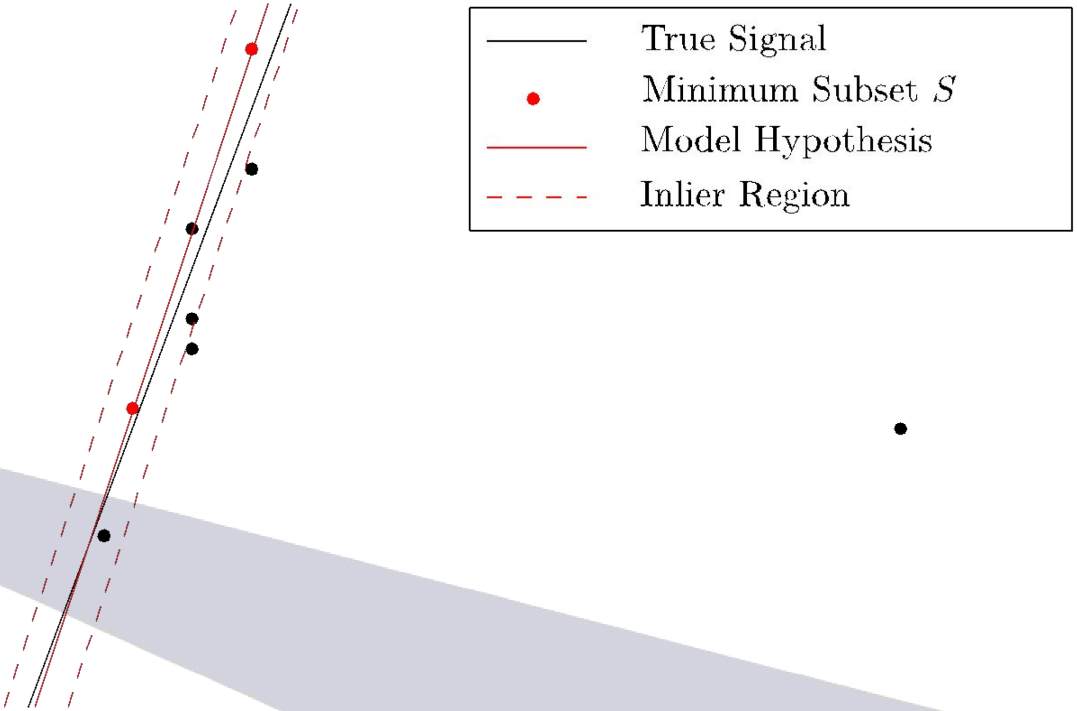
Random Sample Consensus (RANSAC)



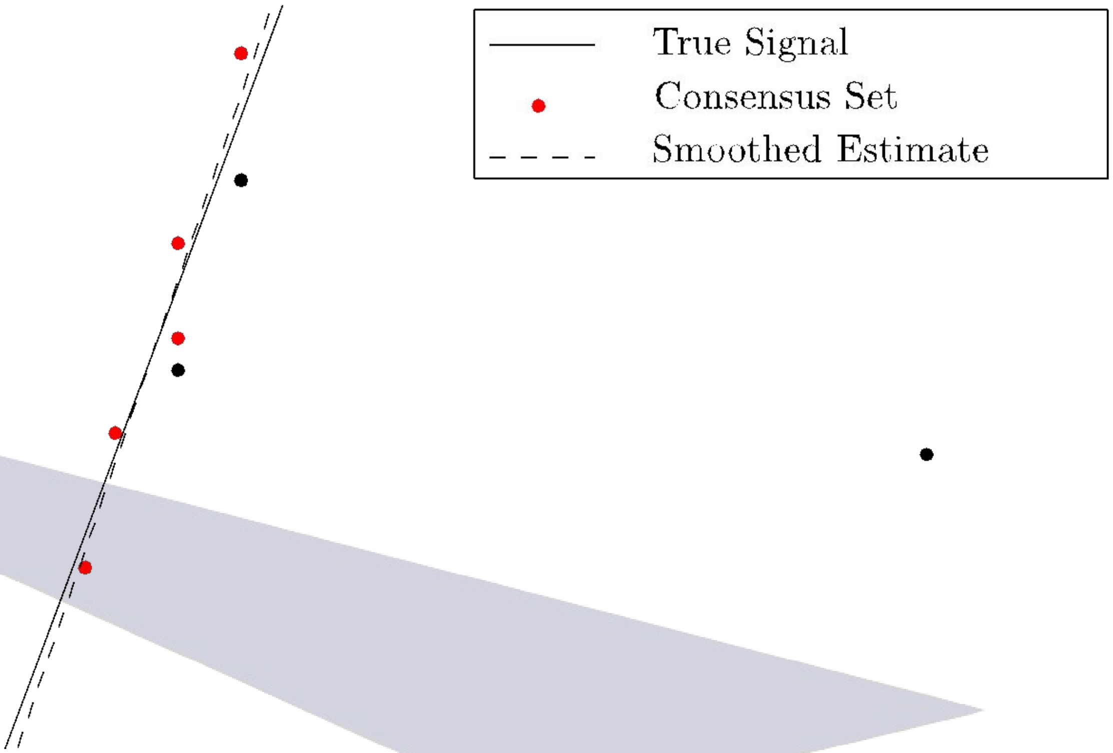
Random Sample Consensus (RANSAC)



Random Sample Consensus (RANSAC)

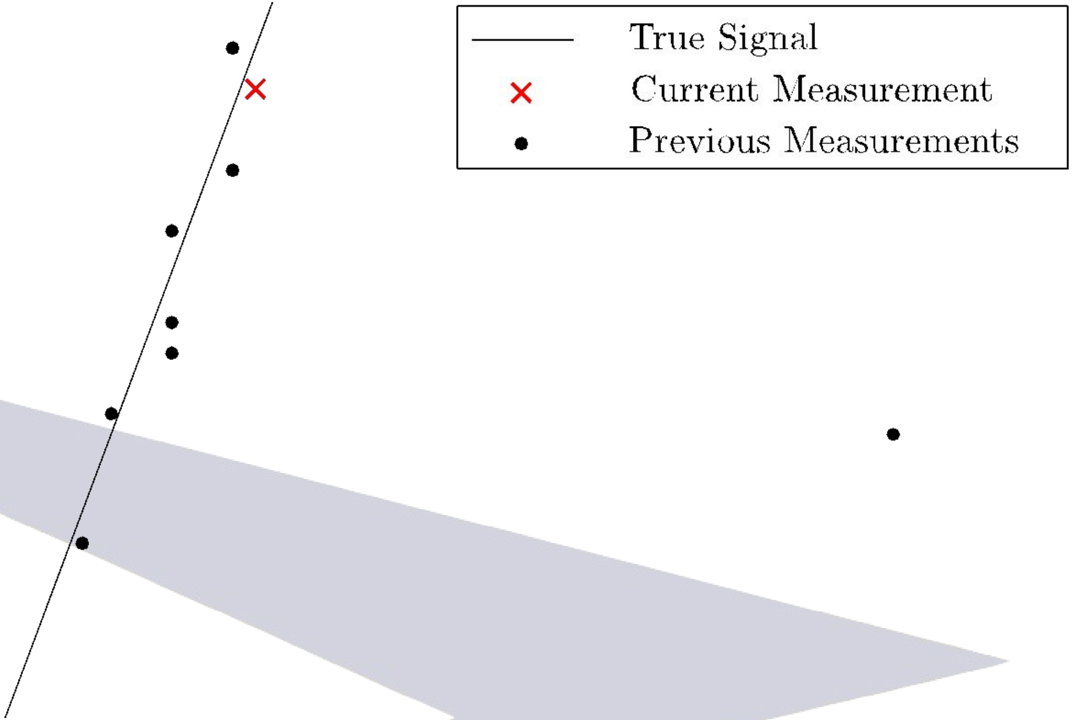


Random Sample Consensus (RANSAC)



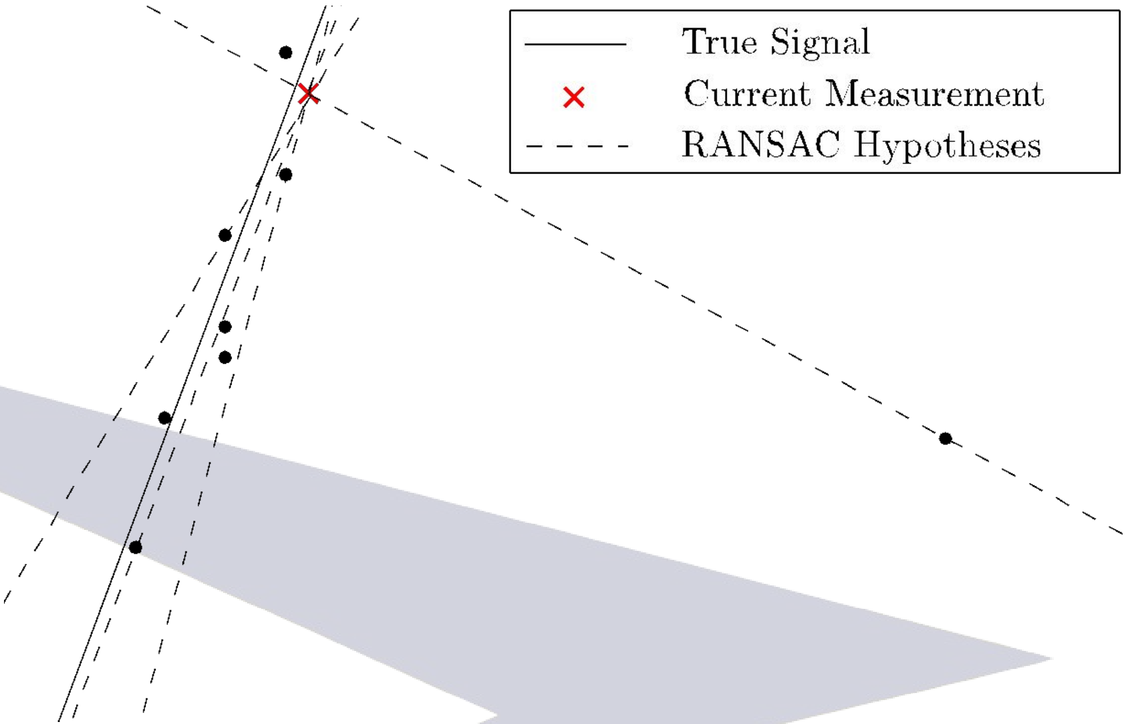
Recursive RANSAC

Measurements received sequentially:



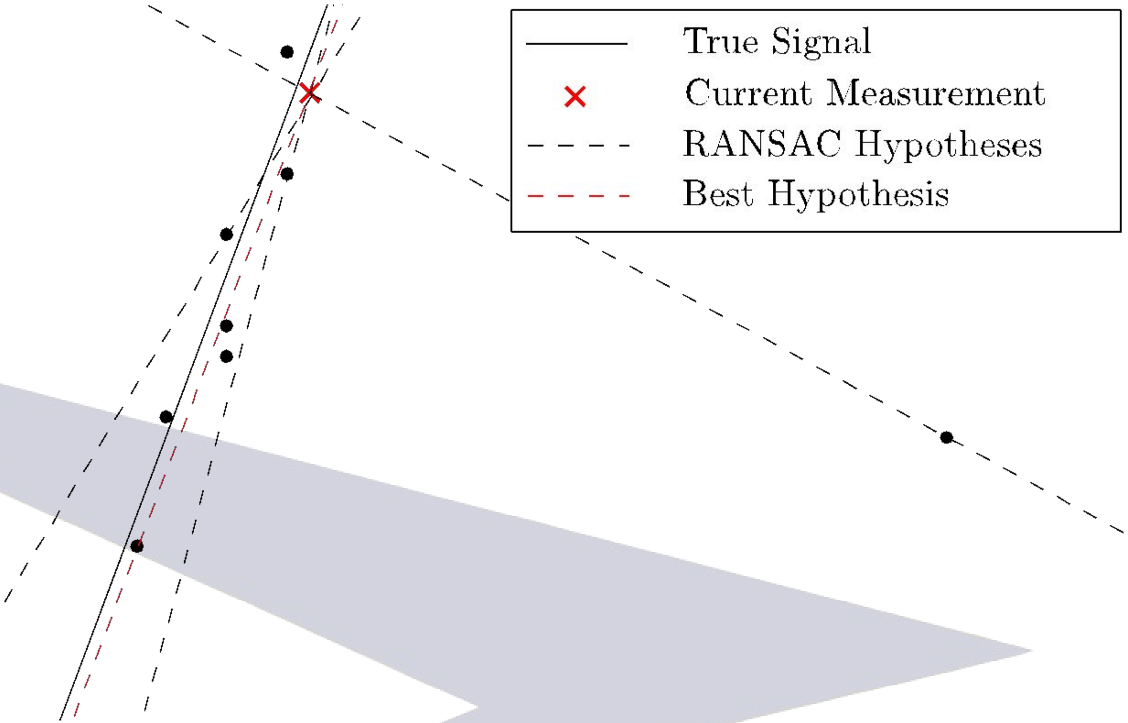
Recursive RANSAC

Use current measurement with RANSAC to form potential models / hypothesis:



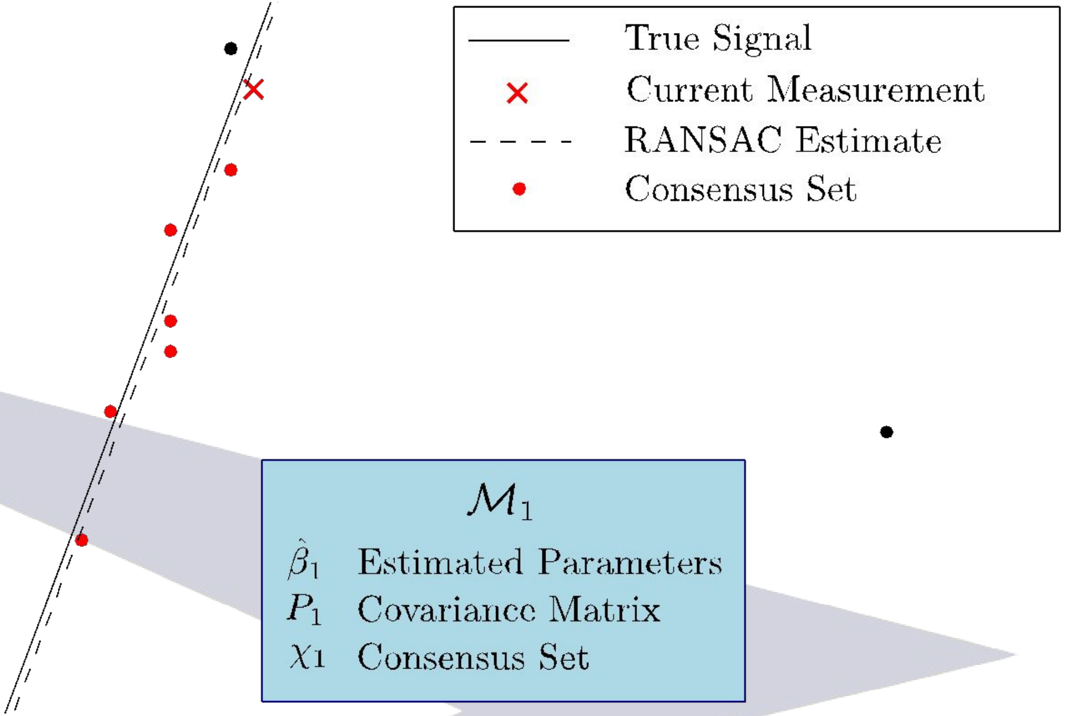
Recursive RANSAC

Use best hypothesis to form $\mathcal{M}_1$:



Recursive RANSAC

Use current measurement with RANSAC to form a model:

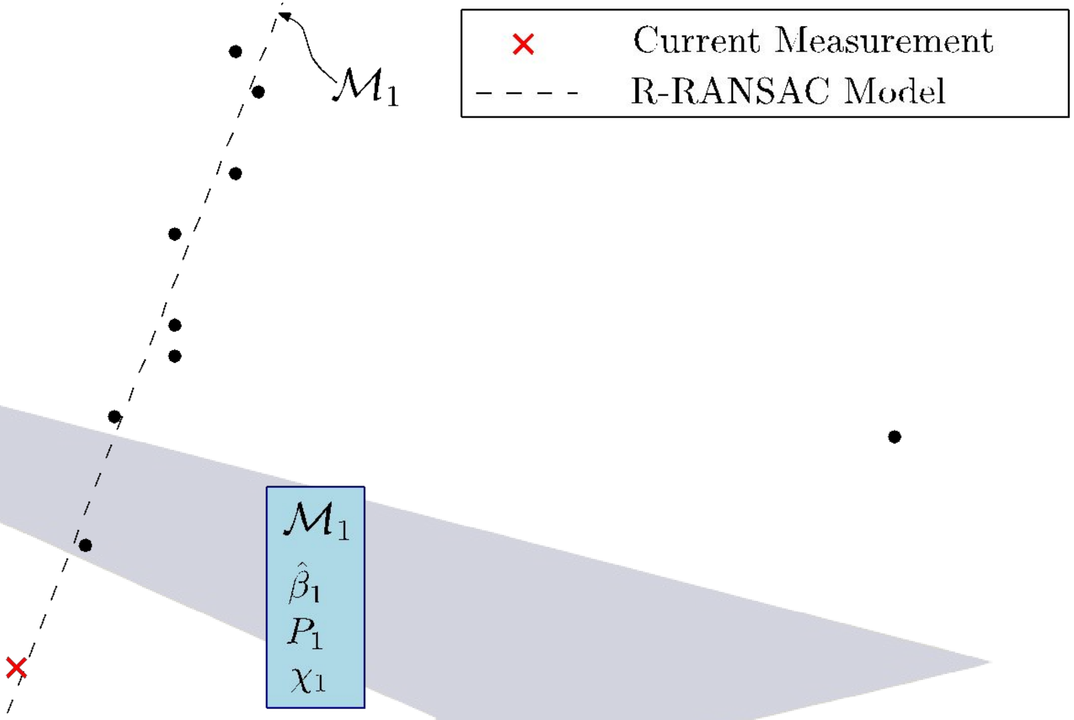


$\mathcal{M}_1$

$\hat{\beta}_1$	Estimated Parameters
P_1	Covariance Matrix
χ_1	Consensus Set

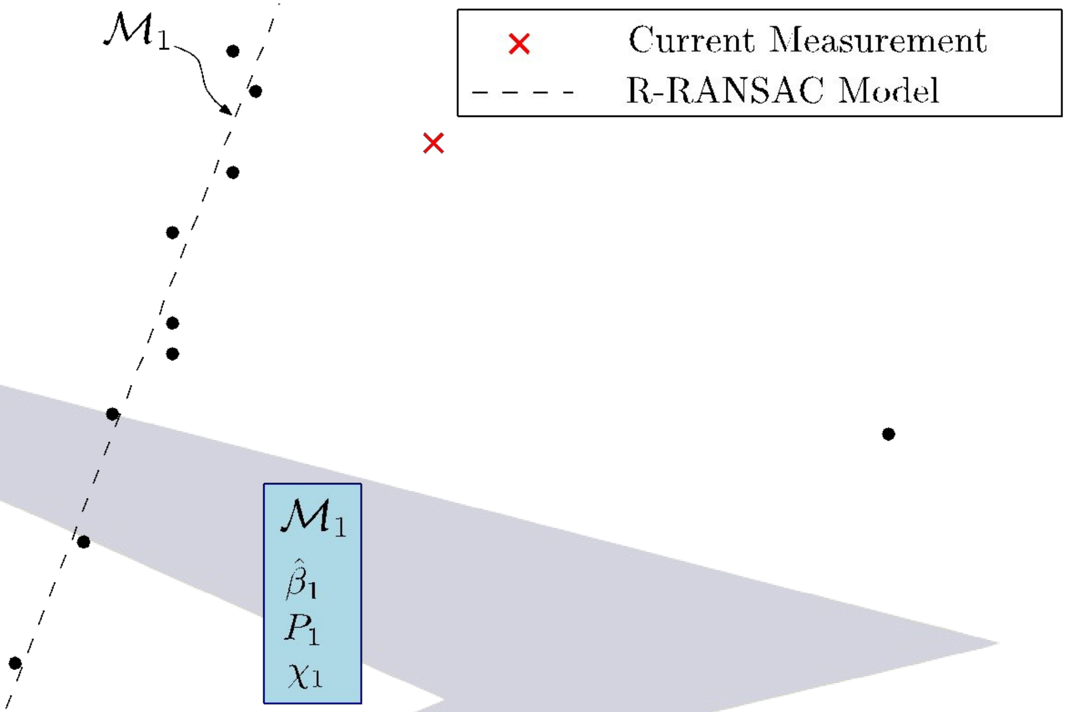
Recursive RANSAC

Next Measurement: Inlier to model $\mathcal{M}_1$



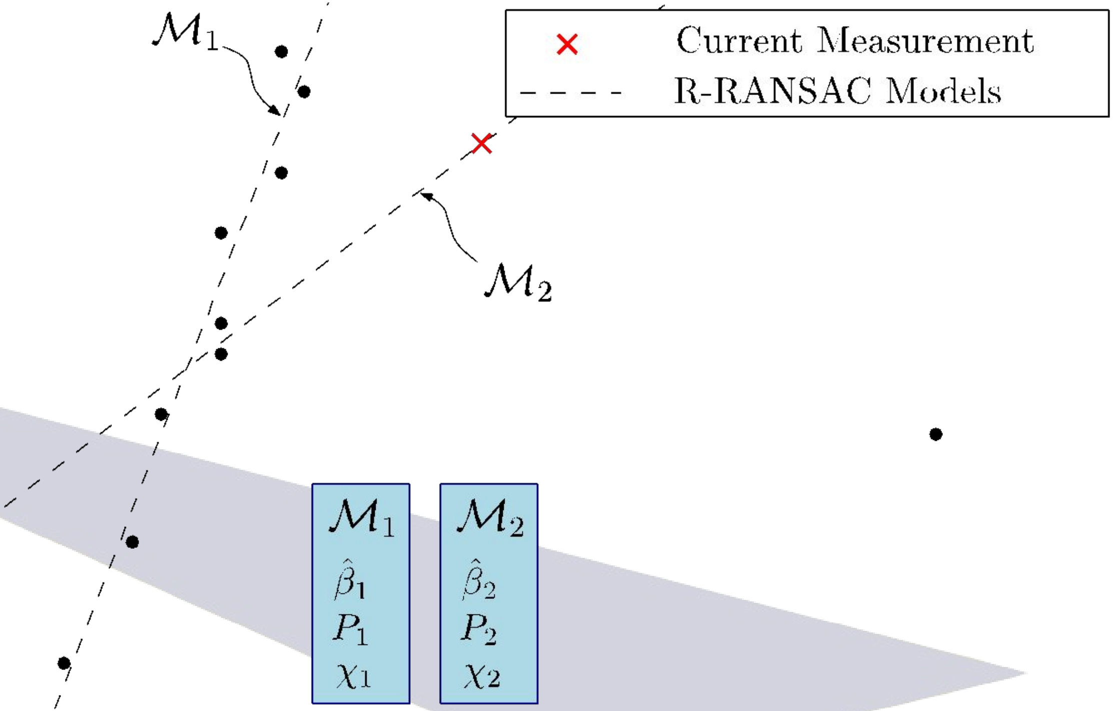
Recursive RANSAC

Next Measurement: Outlier to model $\mathcal{M}_1$.
Is it an outlier or does it come from another model?



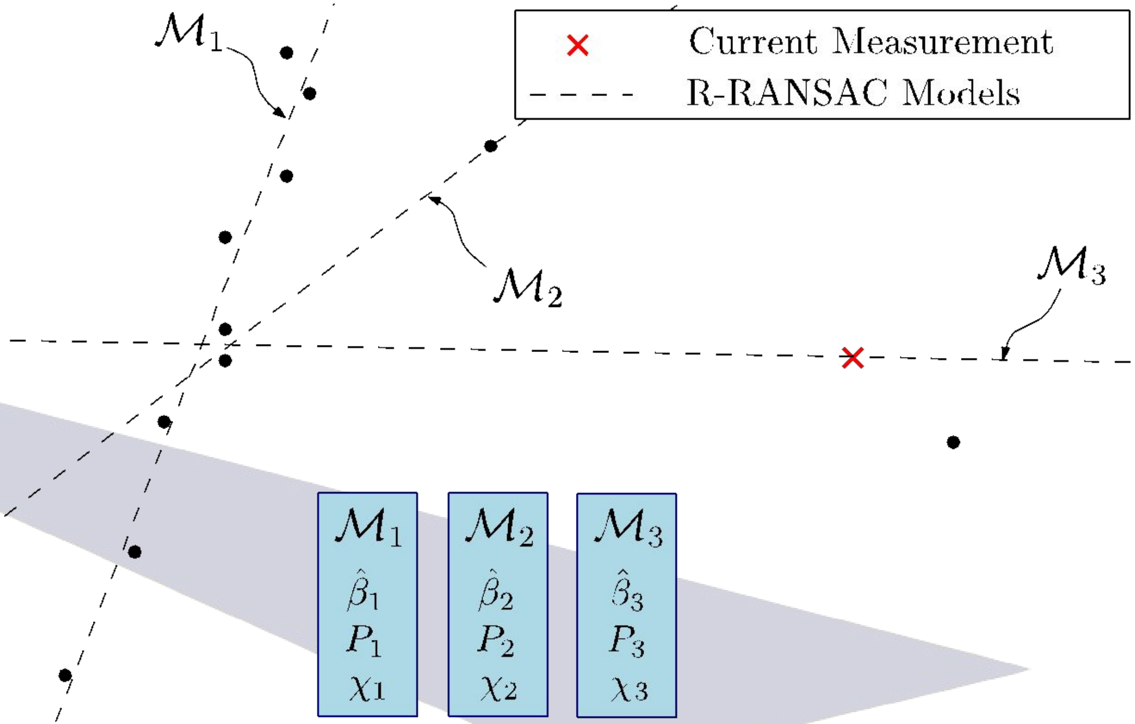
Recursive RANSAC

Next Measurement: Outlier to model $\mathcal{M}_1$


$\mathcal{M}_1$
$\mathcal{M}_2$
✕ Current Measurement
- - - - R-RANSAC Models
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2

Recursive RANSAC

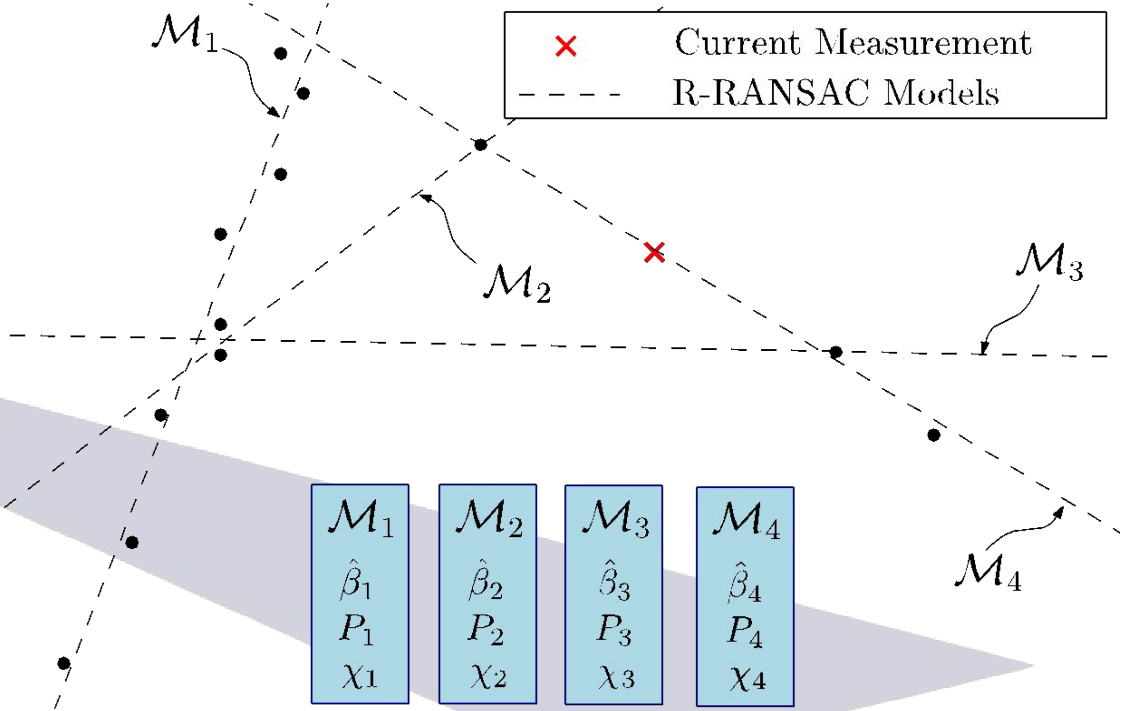
Next Measurement: Outlier to models $\mathcal{M}_1$ and $\mathcal{M}_2$



$\mathcal{M}_1$
$\mathcal{M}_2$
$\mathcal{M}_3$
× Current Measurement
- - - - R-RANSAC Models
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2
$\mathcal{M}_3$ $\hat{\beta}_3$ P_3 χ_3

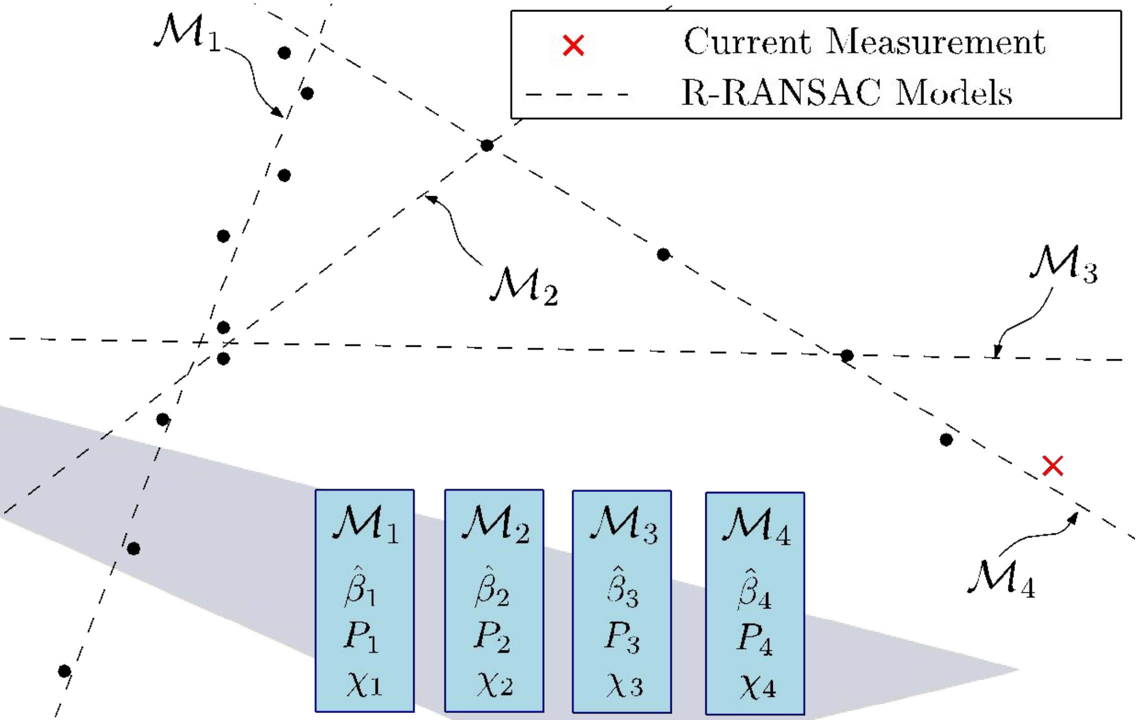
Recursive RANSAC

Next Measurement: Outlier to model $\mathcal{M}_1$–$\mathcal{M}_3$


×
Current Measurement
- - - -
R-RANSAC Models
$\mathcal{M}_1$
$\mathcal{M}_2$
$\mathcal{M}_3$
$\mathcal{M}_4$
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2
$\mathcal{M}_3$ $\hat{\beta}_3$ P_3 χ_3
$\mathcal{M}_4$ $\hat{\beta}_4$ P_4 χ_4

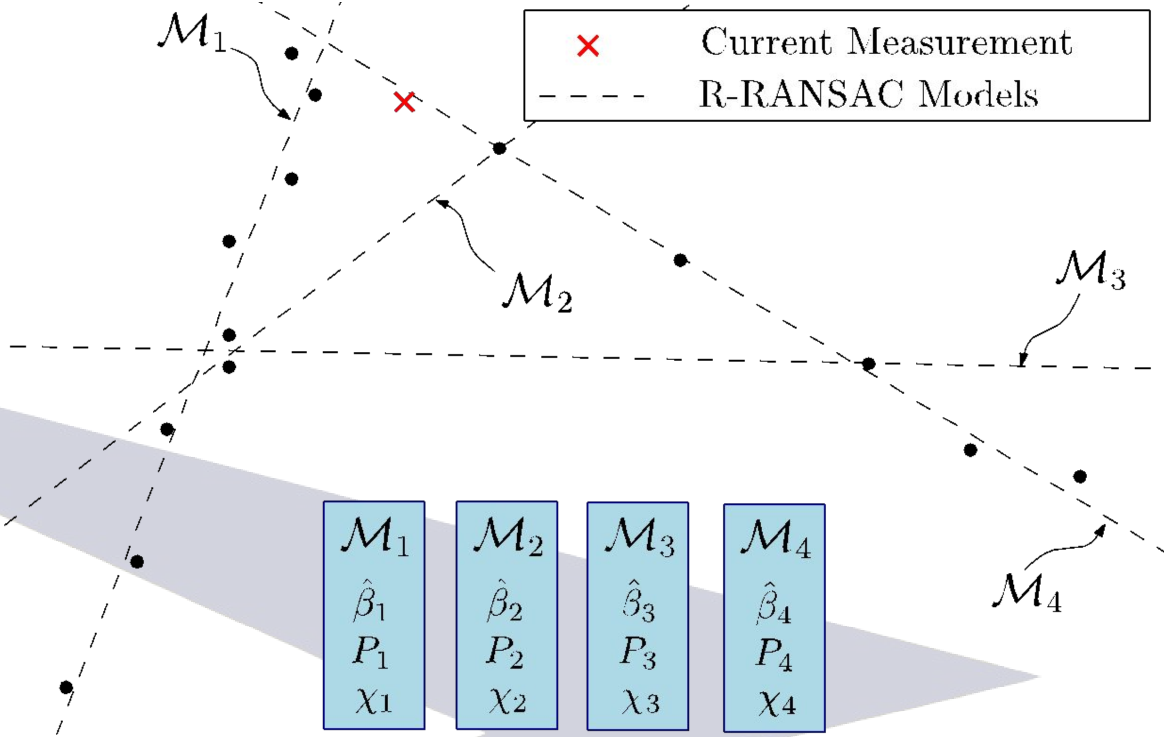
Recursive RANSAC

Next Measurement: Inlier to model $\mathcal{M}_4$


$\mathcal{M}_1$
× Current Measurement
- - - - R-RANSAC Models
$\mathcal{M}_2$
$\mathcal{M}_3$
$\mathcal{M}_4$
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2
$\mathcal{M}_3$ $\hat{\beta}_3$ P_3 χ_3
$\mathcal{M}_4$ $\hat{\beta}_4$ P_4 χ_4

Recursive RANSAC

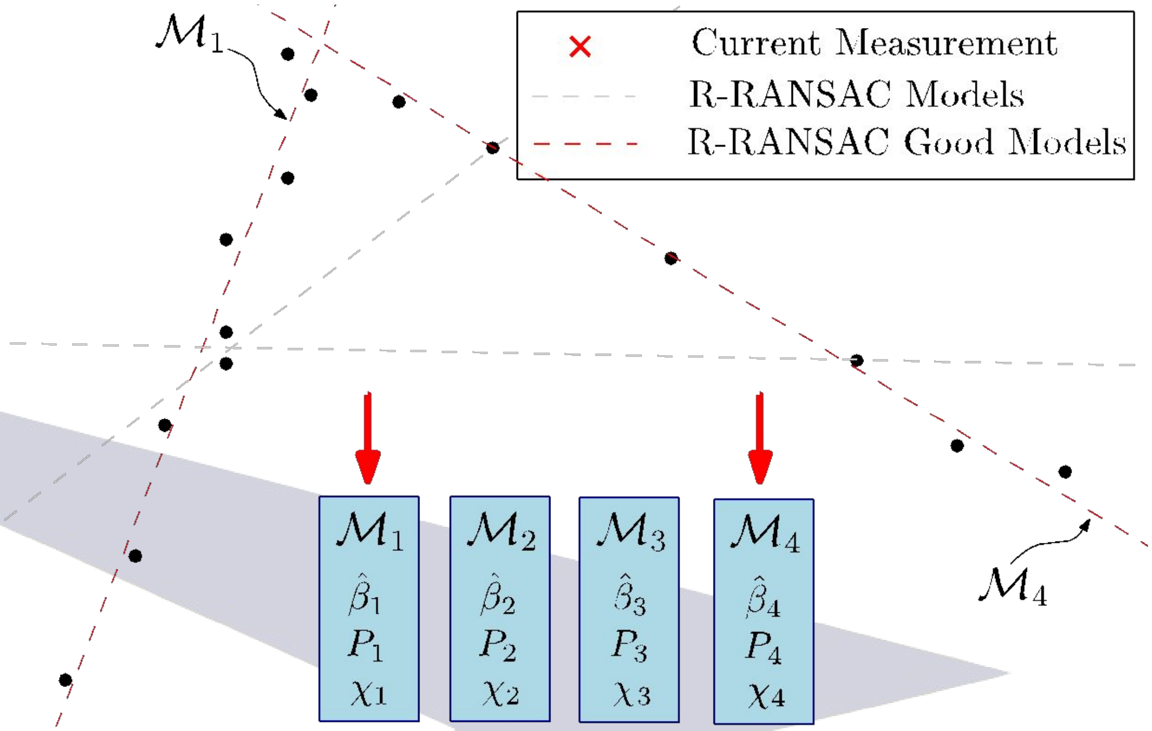
Next Measurement: Inlier to model $\mathcal{M}_4$



× Current Measurement
- - - - R-RANSAC Models
$\mathcal{M}_1$
$\mathcal{M}_2$
$\mathcal{M}_3$
$\mathcal{M}_4$
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2
$\mathcal{M}_3$ $\hat{\beta}_3$ P_3 χ_3
$\mathcal{M}_4$ $\hat{\beta}_4$ P_4 χ_4

Recursive RANSAC

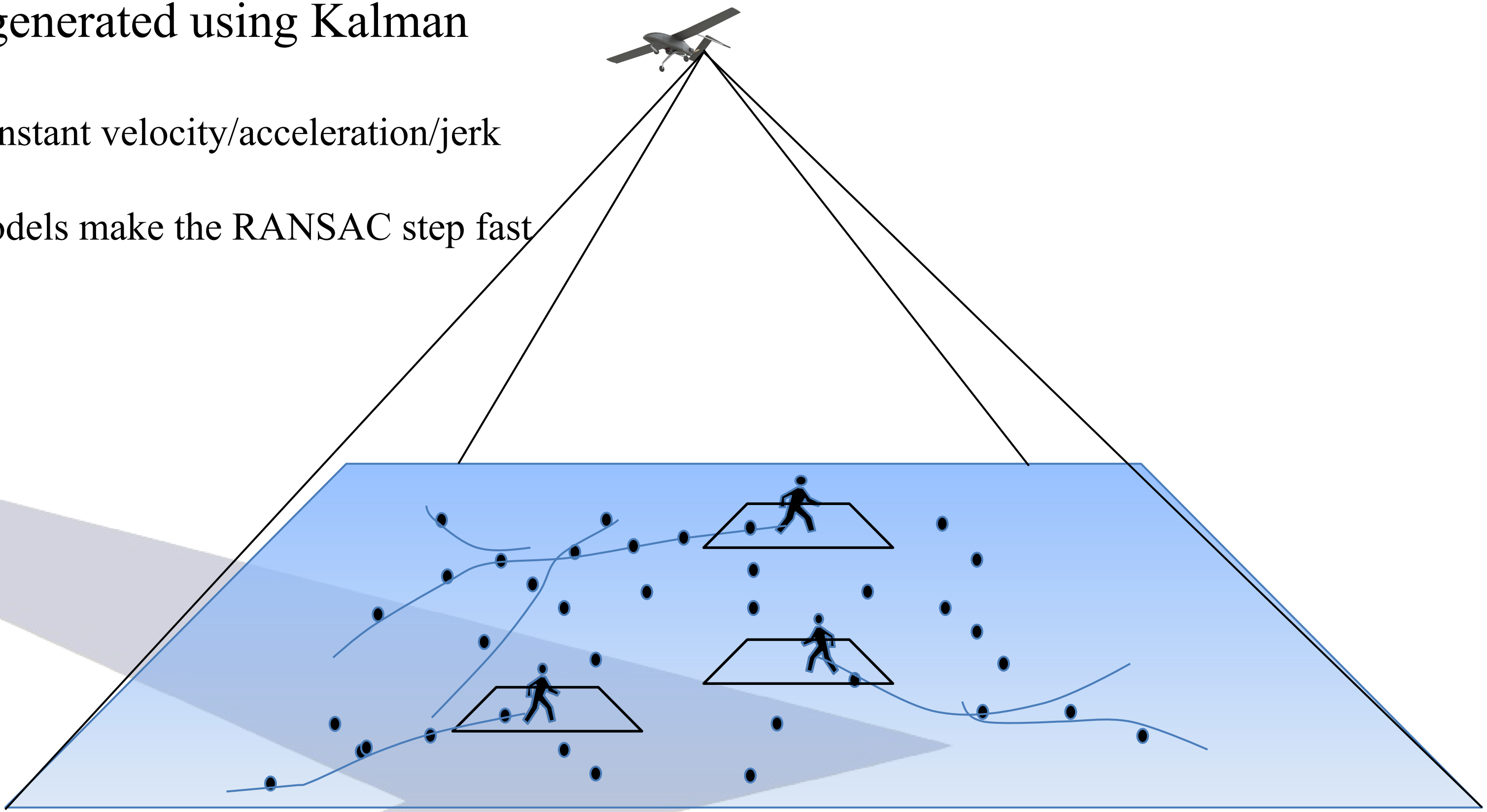
Identify models with most inliers: Prune $\mathcal{M}_2$ and $\mathcal{M}_3$.



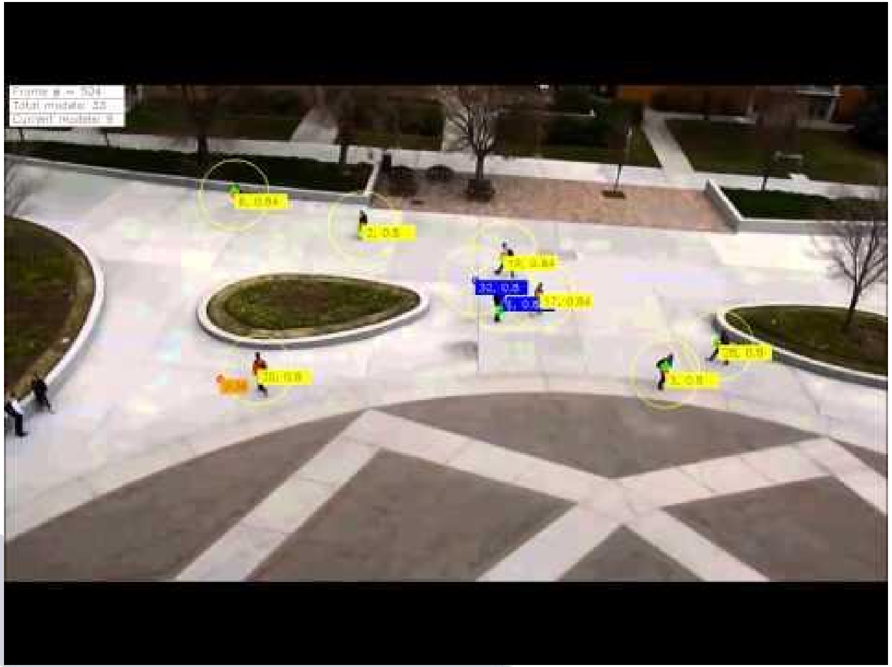
$\mathcal{M}_1$
× Current Measurement
R-RANSAC Models
R-RANSAC Good Models
$\mathcal{M}_1$ $\hat{\beta}_1$ P_1 χ_1
$\mathcal{M}_2$ $\hat{\beta}_2$ P_2 χ_2
$\mathcal{M}_3$ $\hat{\beta}_3$ P_3 χ_3
$\mathcal{M}_4$ $\hat{\beta}_4$ P_4 χ_4
$\mathcal{M}_4$

Recursive RANSAC for Dynamic Tracking

- Models are generated using Kalman Filter
  - Nearly constant velocity/acceleration/jerk models
  - Linear models make the RANSAC step fast



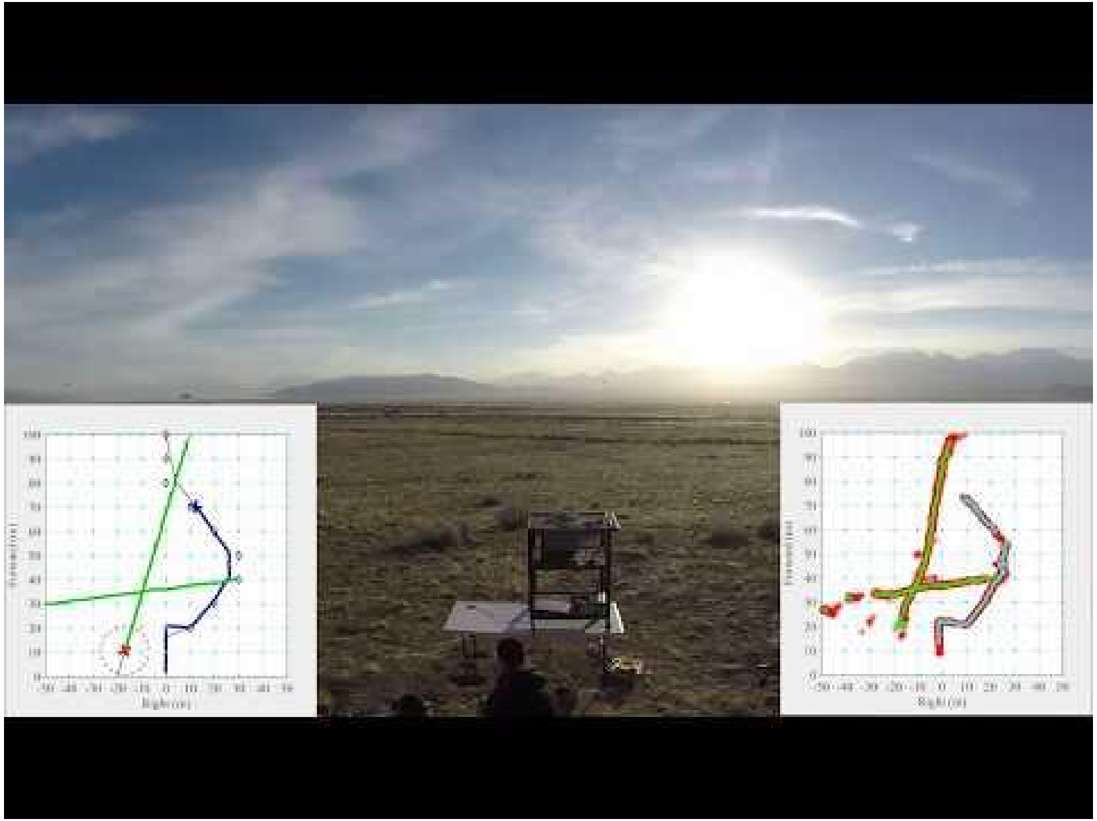
R-RANSAC Applied to Video Tracking



- IMM R-RANSAC with PDA data association
  - Nearly-constant acceleration models with three levels of process noise
- C/C++ Implementation

Moving Camera Low Light

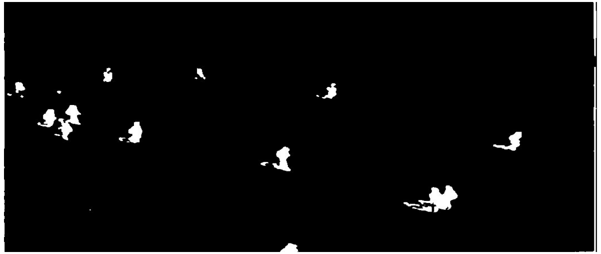
Radar Tracking using R-RANSAC



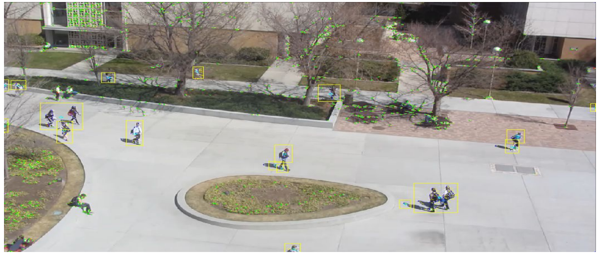
Moving Object w/ Moving Camera

- The first step in robust tracking is robust object detection.
- We use widely-known computer vision techniques (Lucas-Kanade Optical Flow, RANSAC Homography estimation) to register sequential frames and detect potential targets.

Find feature points in the current frame. We currently use Good Features to Track [1]. Using RANSAC, we match the features points to those of the last frame to find an affine homography between frames.



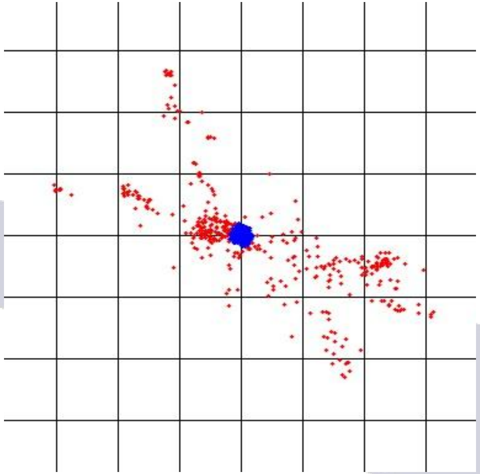
Taking the difference of the registered images gives a mask of possible moving objects.

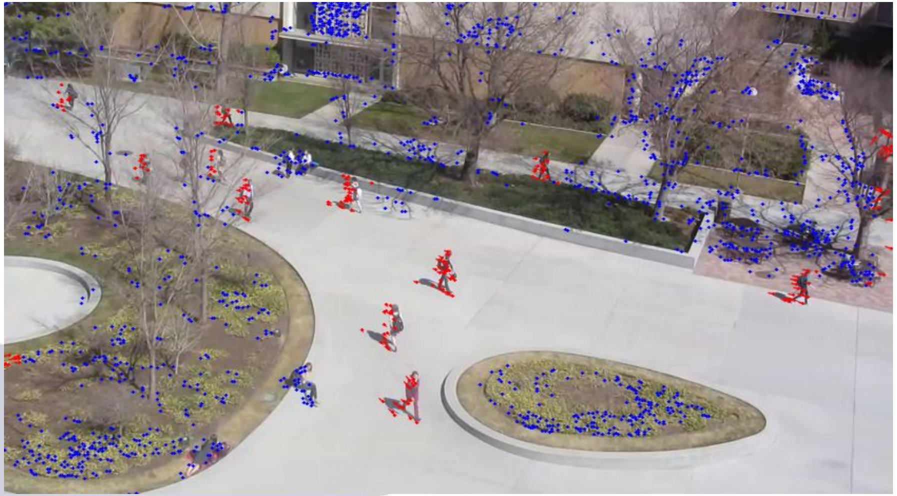


We accept feature points that fall on the white parts of the mask as moving points. These are grouped together by position and velocity to identify individual targets. Velocity information help discriminate measurements from crossing targets.

Moving Object Detection

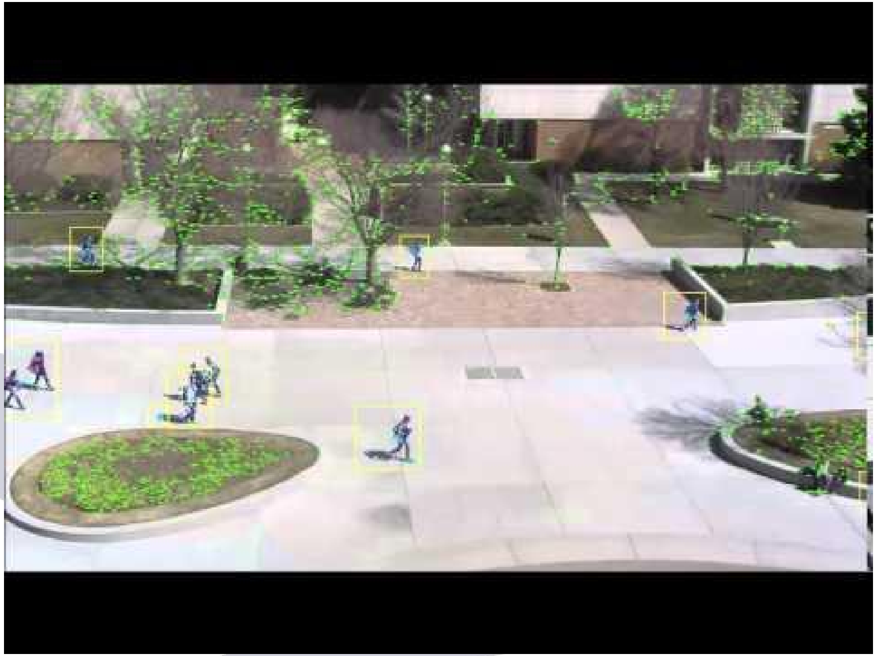
1. Detect features using Good Features to Track (Shi-Tomasi method)
2. Determine pixel movement using Lucas-Kanade Optical Flow
3. Calculate homography using RANSAC
4. Calculate net pixel movement by subtracting camera movement
5. Threshold pixel movement to detect moving objects





Track Management

- After moving points are detected, we input the measurements to the R-RANSAC algorithm to generate, merge, and delete tracks as needed.
- We use a nearly constant jerk model with position and velocity measurement as inputs to the Kalman Filter(s).

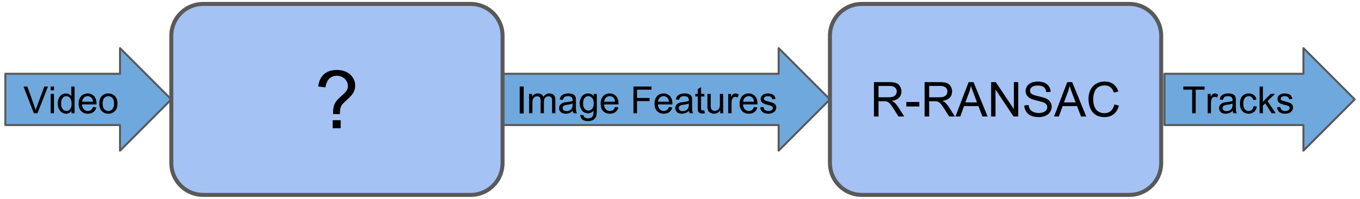


Outline

- Introduction and Motivation
- GPS Degraded Navigation
- Robust Target Tracking
  - Recursive RANSAC
  - Track Continuity
  - Inertial vs image tracking - SLAM
  - Flight control and gimbal servoing
  - Safe2Ditch
  - Hardware implementation

Motivation

We choose what computer vision algorithm to implement.

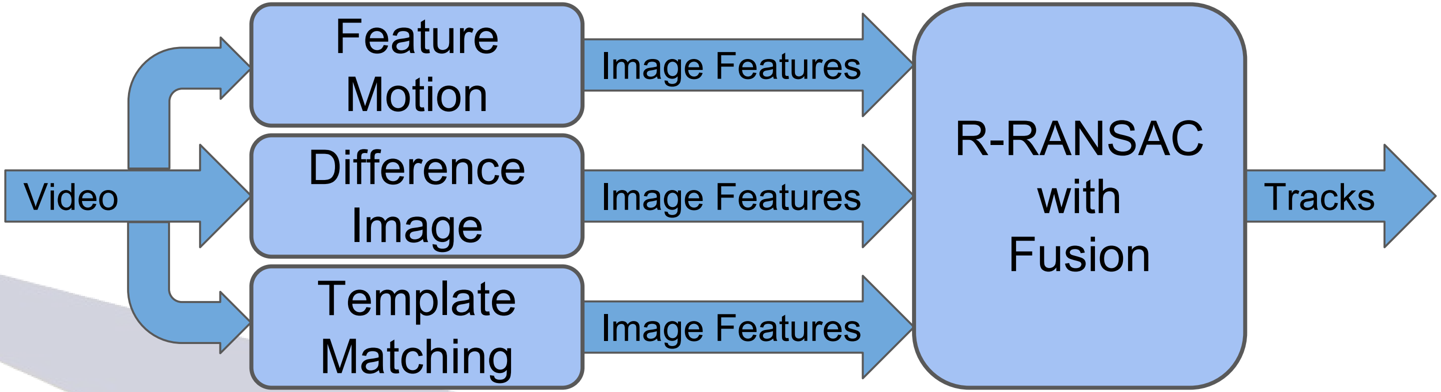


Some of many possible methods (each with advantages and disadvantages):

Feature Motion	Direct velocity information	Less effective for distant targets
Difference Image	Frequent measurements	A bit noisy, position-only
Direct Template Matching	Can track stopped targets	Cannot initialize tracks, drifts

Sensor Fusion in R-RANSAC (new)

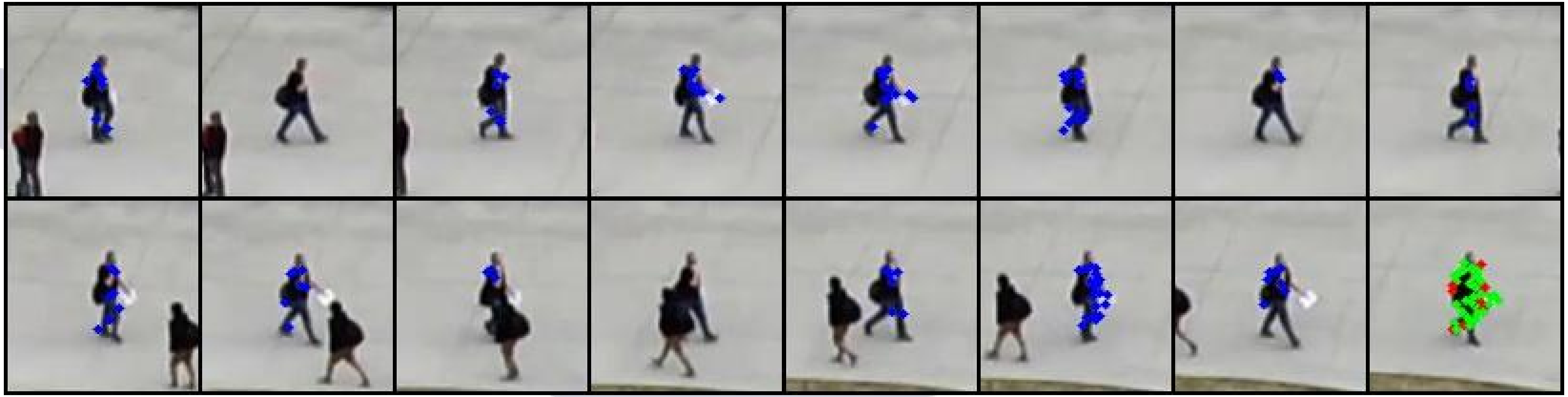
The CV frontend is generally the weak link. So we expanded R-RANSAC to accept multiple measurement sources, leveraging the strengths of each.



Inside R-RANSAC, a centralized fuser combines the provided information into global estimates. This provides fault tolerance in the visual front end and robust track continuity.

Feature-Aided Tracking

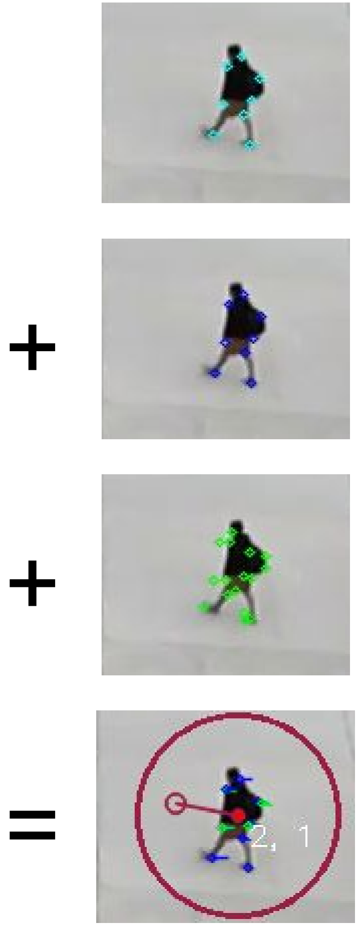
- Motivation: Feature matching has the potential to disambiguate tracks
- Store and match previously-seen features to improve data ass



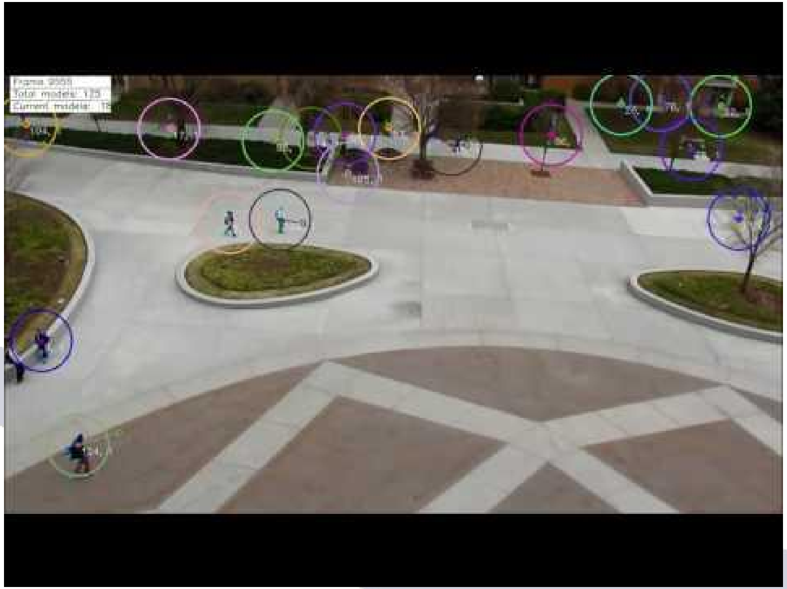
Feature-Aided Tracking Initial Results

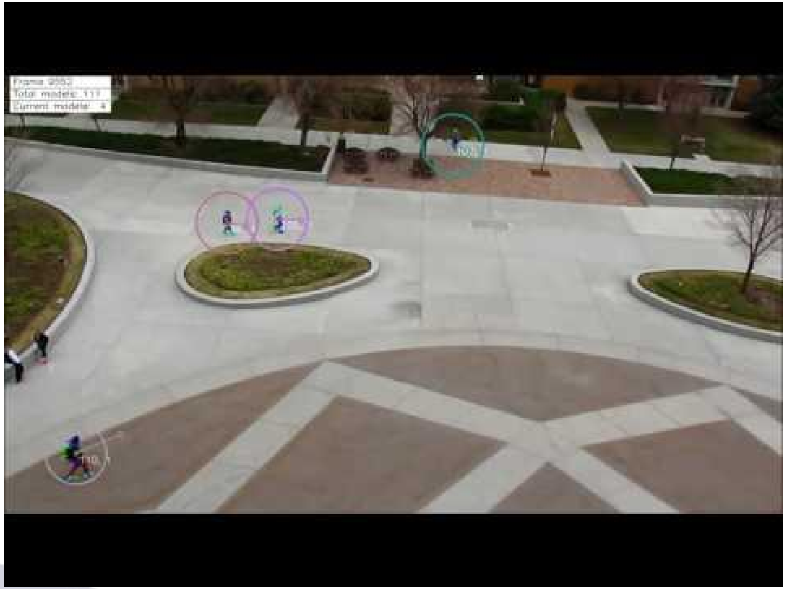
Combining Tracking Methods by Prioritization

- For each area of the image prioritize
  - ◆ 1st: Moving points (reduce drift)
  - ◆ 2nd: Feature matches (appearance model)
  - ◆ 3rd: Optical flow (stopped objects)
- Result: One valid measurement for each area of the image

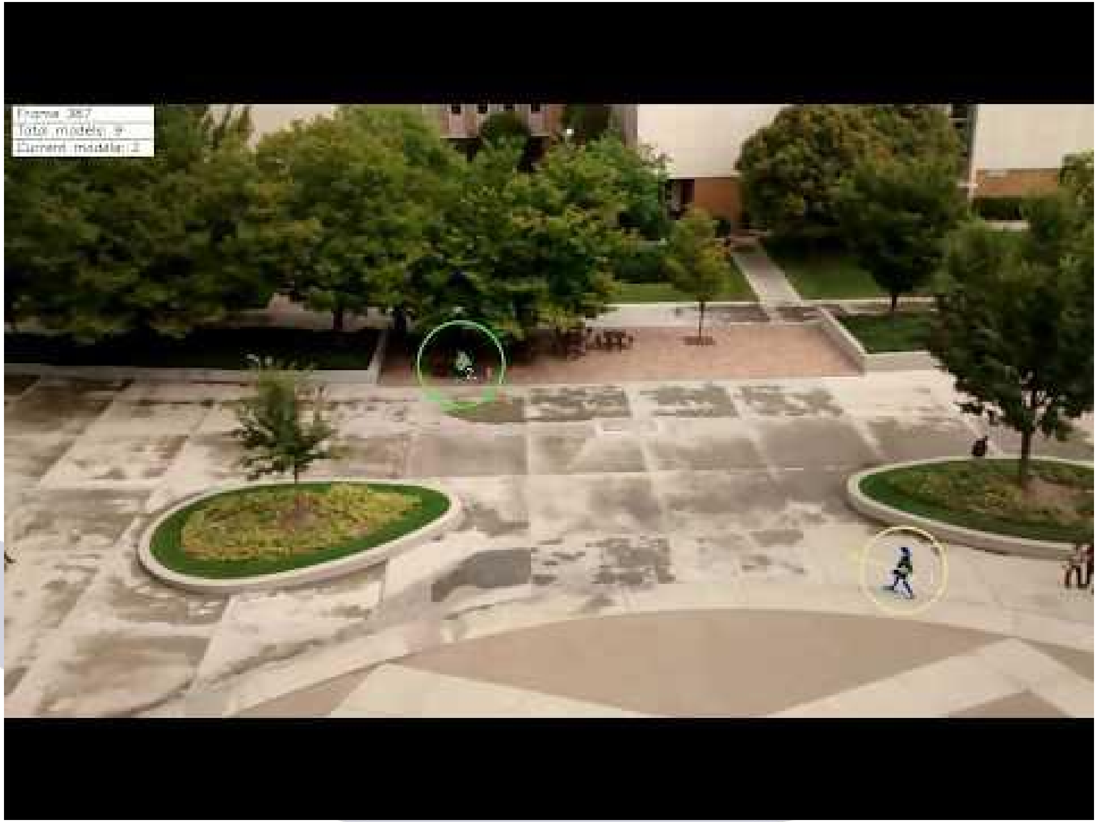


Ghost Track Reduction Results





Results on a Moving Camera



Results: Before and After (new)

Poor Video Conditions:

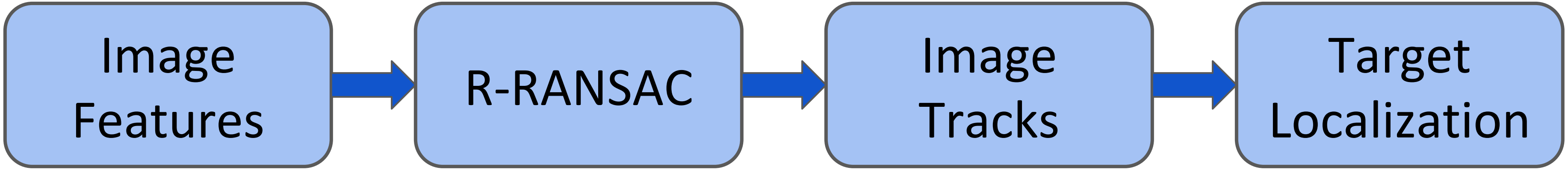
- Aggressive UAV motion, no gimbal
- Rolling shutter and compression
- No GPU (for this case)

Outline

- Introduction and Motivation
- GPS Degraded Navigation
- Robust Target Tracking
  - Recursive RANSAC
  - Track Continuity
  - Inertial vs image tracking - SLAM
  - Flight control and gimbal servoing
  - Safe2Ditch
  - Hardware implementation

Tracking Framework

- Challenge: object tracking in image frame versus inertial frame
- Method 1: Image-based tracking



- Method 2: Inertial-based tracking

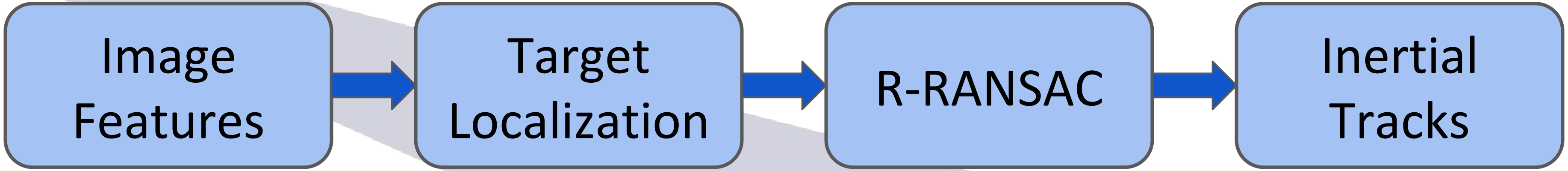
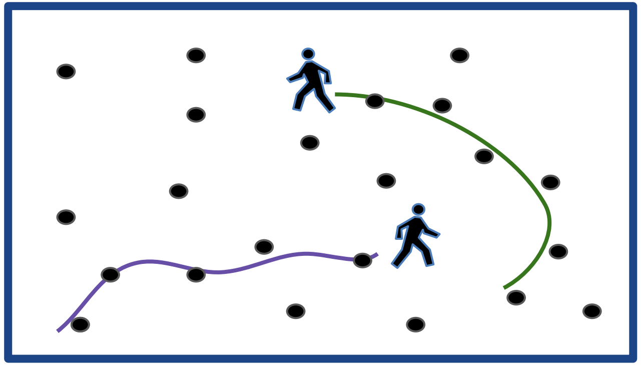


Image vs Inertial Tracking

Image-based Tracking

- Measurements have little variance
- Gimbal pointing reduces image movement, causing lost tracks
- Image is easily cluttered with many targets



Inertial-based Tracking

- Large potential variance due to gimbal/uav pose estimation uncertainty
- Readily available tracks for cooperative tracking and control
- Not directly affected by poor homography estimates

Target Localization Challenges

- Small error in UAV/gimbal pose estimates cause large deviations in target localization estimates
- Camera and state estimator must be well synchronized
- Common to have a flat-earth assumption

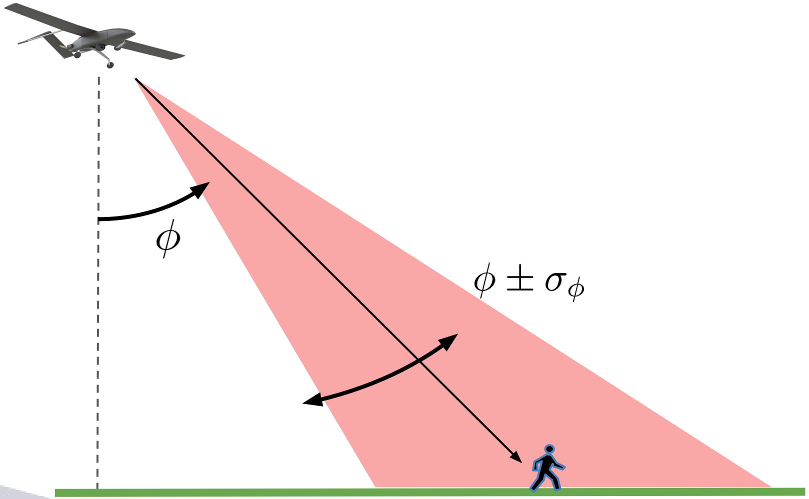
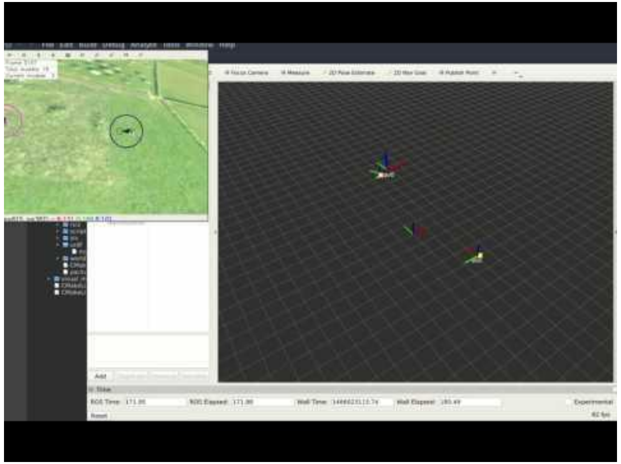


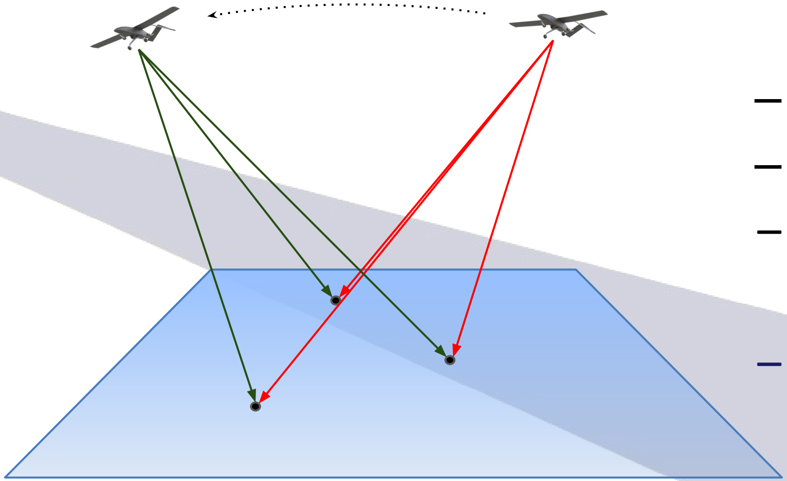
Image Tracking w/ Target Localization

- Gazebo-ROS simulation
- Left image is R-RANSAC tracking two ATVs viewed from a UAV
- Right image is Rviz showing the instantaneous target localization estimate as yellow squares
- Sensors are loosely synchronized (localization estimate diverges on fast rotations)



EKF Target Localization

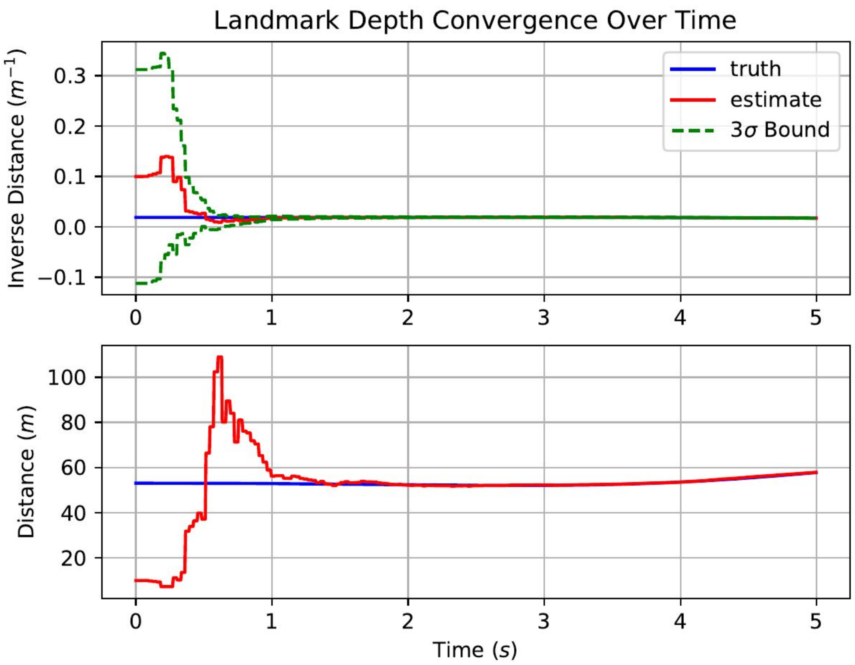
- Robust Visual Inertial Odometry (ROVIO)
  - State-of-the-art EKF-SLAM approach to VO that yields little drift in pose estimation



- Estimates bearing and distance of stationary landmarks relative to the UAV
- Operates in a robocentric framework
- Fuses image data with IMU data
- With landmark locations known, target can be located relative to them
- We separate UAV state from landmark state

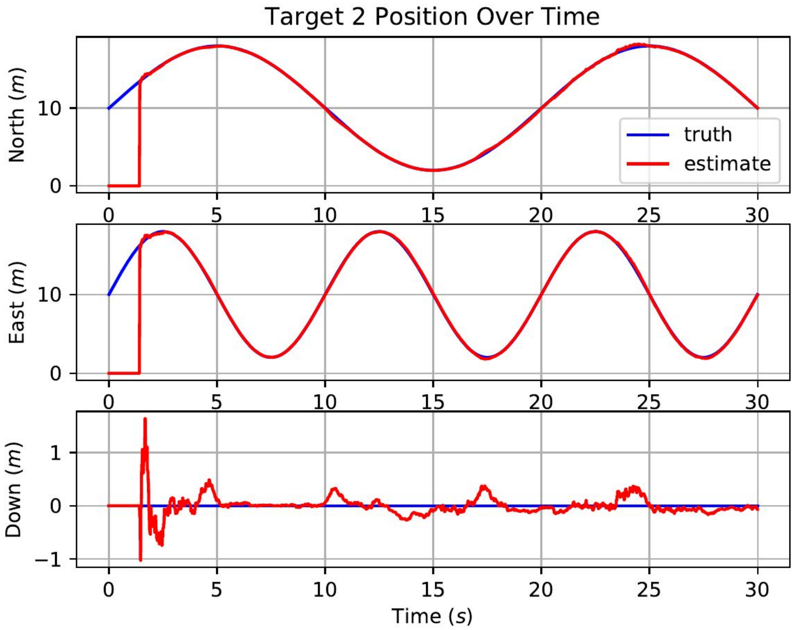
EKF Target Localization

- Simulation results of EKF estimator on a quadrotor
  - Landmark and UAV estimates converge quickly
  - Poor initializations lead to instability
  - A globally stable solution, possibly less accurate, would be preferable



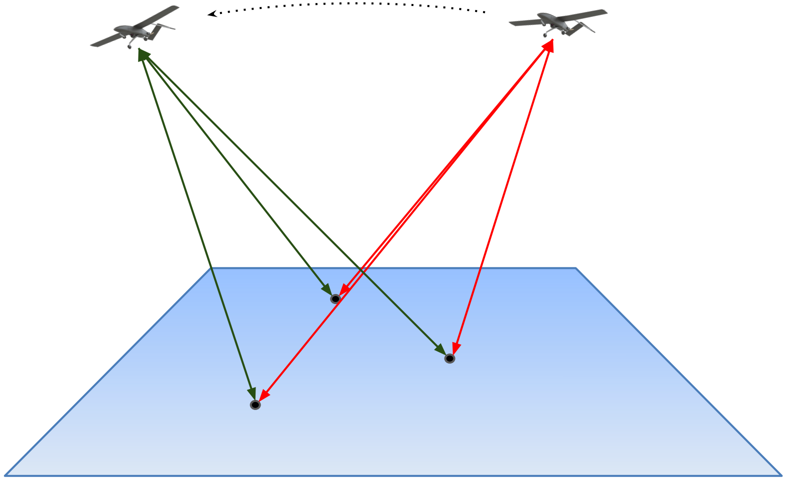
EKF Target Localization

- Once 3 or more landmark estimates have converged:
  - Approximate a plane via least squares using landmark estimates
  - Inertial target location is approximately the intersection of the target's bearing vector and this newly approximated plane
  - Results shown to the right



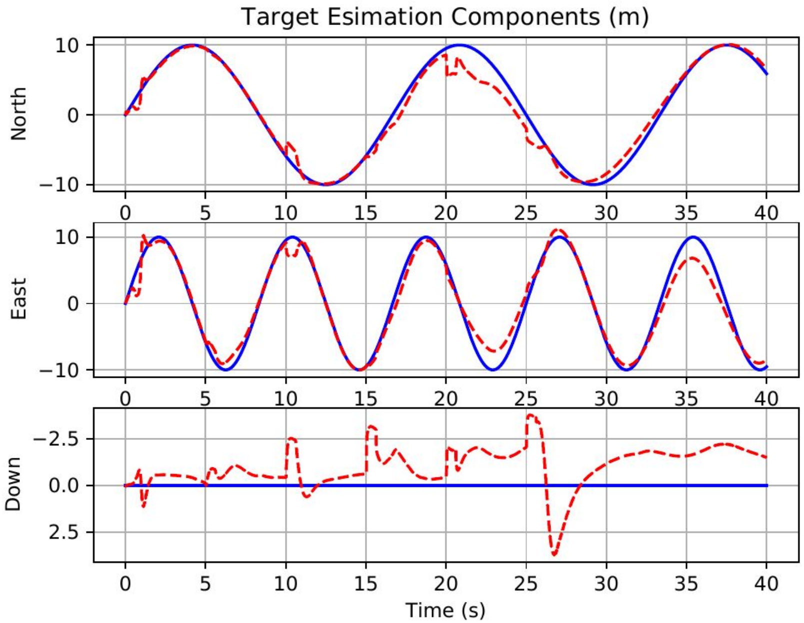
NO-SLAM Target Localization

- Nonlinear Observer SLAM
- Simultaneously estimates UAV position, velocity bias, and landmark inertial positions
- Convergence may be slow, iterated observer improves this
- Assumes attitude is known
- Guaranteed global stability iff:
  - Origin direction is measured
  - Bearings persistently excited



NO-SLAM Target Localization

- Results of target estimation to the right
- Converges quickly after each new origin due to observer iteration
- Error due to imperfect origin estimates and deteriorating attitude estimation



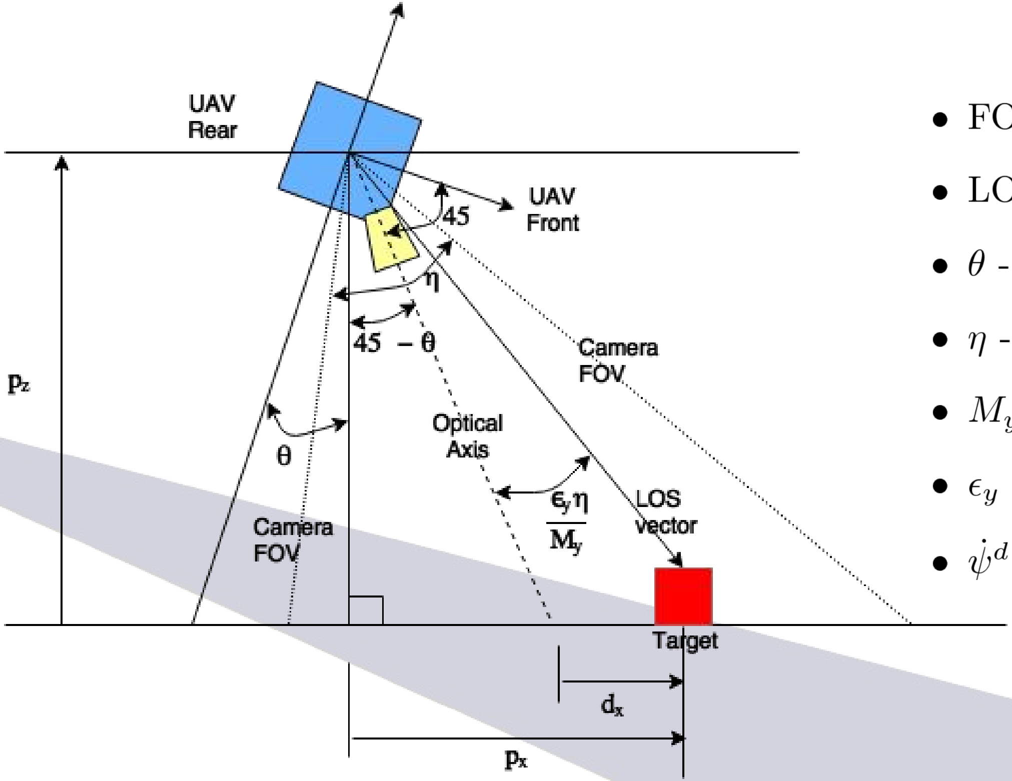
Next Steps

- Smooth estimation after new origin
- Thoroughly analyze observer iteration
- Improve NO-SLAM to work for hover and constant velocity flights

Outline

- ➢ Introduction and Motivation
- ➢ GPS Degraded Navigation
- ➢ Robust Target Tracking
  - – Recursive RANSAC
  - – Track Continuity
  - – Inertial vs image tracking - SLAM
  - – Flight control and gimbal servoing
  - – Safe2Ditch
  - – Hardware implementation

Current Progress - Flight Control



- FOV - camera field of view
- LOS - line of sight vector from UAV to target
- θ - UAV pitch angle in degree
- η - camera vertical angle of view in degree
- M_y - (vertical pixel resolution)/2 in pixel
- ϵ_y - target location on vertical axis in image in pixel
- $\dot{\psi}^d$ - desired yawrate

Current Progress - Flight Control

- UAV forward motion

$$p_x = p_z \tan(45^\circ - \theta - \frac{\epsilon_y \eta}{M_y})$$

$$d_x = p_z(\tan(45^\circ - \theta - \frac{\epsilon_y \eta}{M_y}) - \tan(45^\circ - \theta))$$

- UAV side motion
  Transform the unit LOS vector from optical frame into Vehicle-1 frame.

$$\begin{bmatrix} \epsilon_x \\ \epsilon_y \\ 1 \end{bmatrix}_{optical frame} \rightarrow \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}_{V-1 frame}$$

$$\dot{\psi}^d = p_y(scale factor)$$

Current Progress - Video

Outline

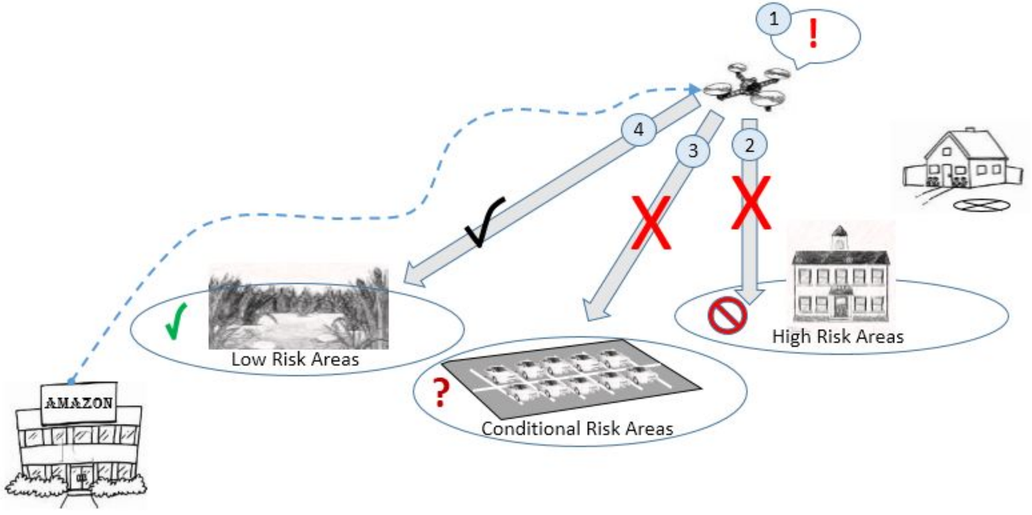
- Introduction and Motivation
- GPS Degraded Navigation
- Robust Target Tracking
  - Recursive RANSAC
  - Track Continuity
  - Inertial vs image tracking - SLAM
  - Flight control and gimbal servoing
  - Safe2Ditch
  - Hardware implementation

Safe2Ditch

- Autonomous safety system designed to enable emergency landings in semi-populated areas
- Uses a fixed camera to detect motion in the landing zone
- Joint project with NASA Langley Research Center

Safe2Ditch

- Autonomously choose a low-risk area to land from database
- Use camera to refine landing zone during descent
- Assumes **60 seconds** for landing

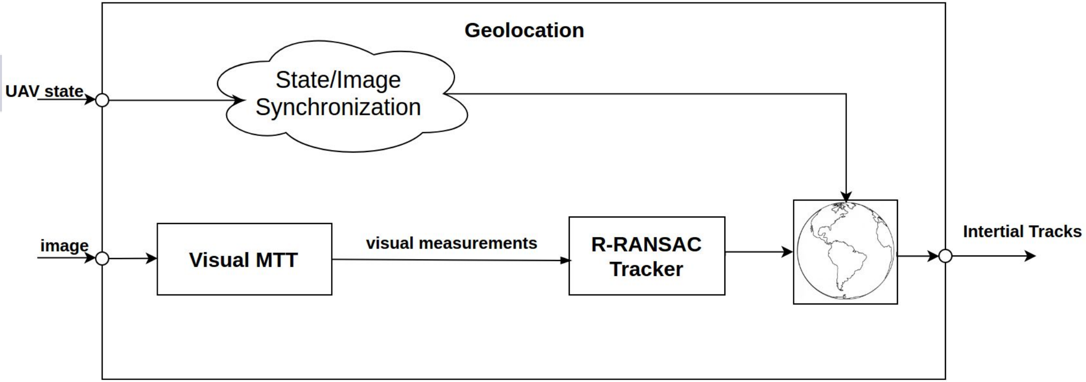


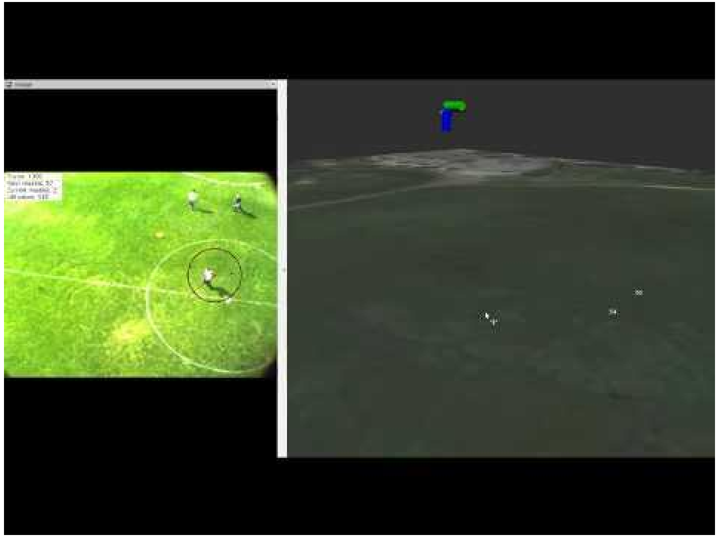
Hardware Platform and Integration

- 3DR Y6 frame
- 3DR GPS + Compass
- Pixhawk + ArduCopter
- NVIDIA Jetson TX2
  - CUDA-enabled Visual MTT
- IDS uEye 800x600 @30fps
- 12mm lens, ~34° AFOV

Geolocation of R-RANSAC Tracks

- Flat-earth model
- ROS hardware implementation
- Creates inertial R-RANSAC tracks

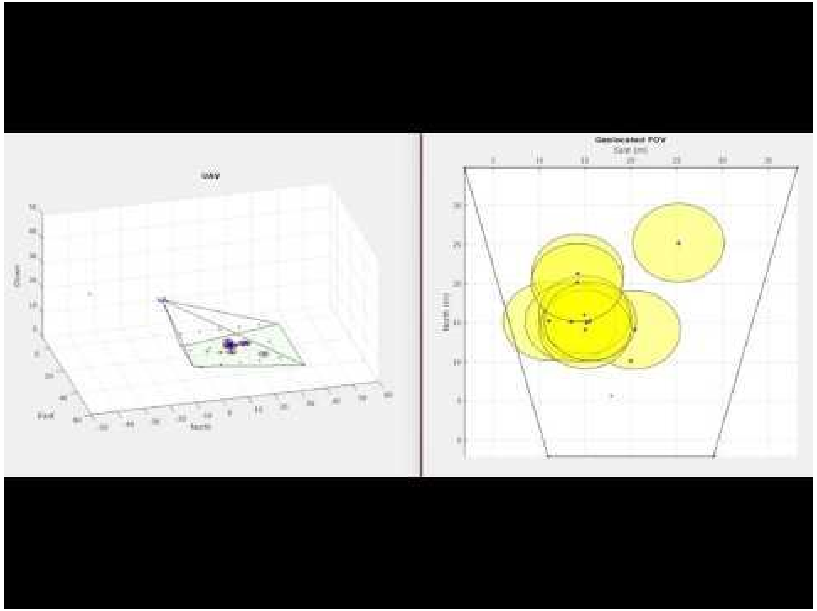




NASA Flight Tests

Potential Field Track Avoidance

- Find the inertial position of the camera’s principal point
- As the multirotor descends, use potential fields to push the principle point away from targets

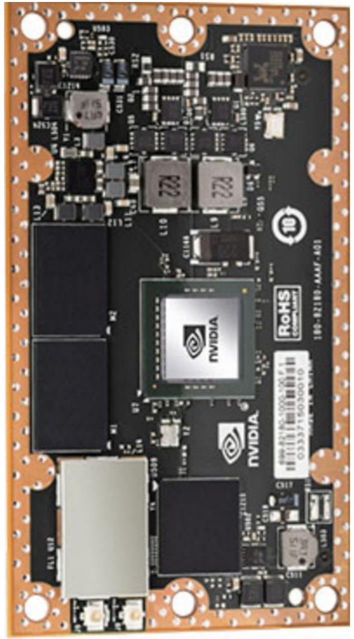


Outline

- Introduction and Motivation
- GPS Degraded Navigation
- Robust Target Tracking
  - Recursive RANSAC
  - Track Continuity
  - Inertial vs image tracking - SLAM
  - Flight control and gimbal servoing
  - Safe2Ditch
  - Hardware implementation

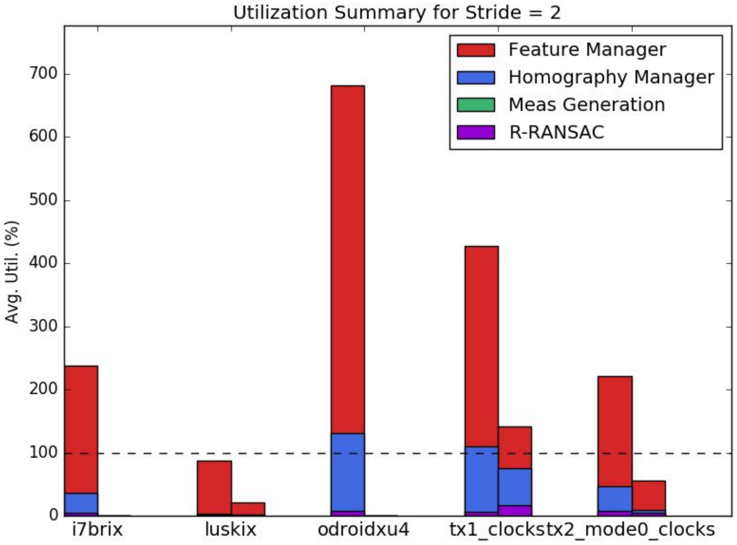
NVIDIA Jetson TX2

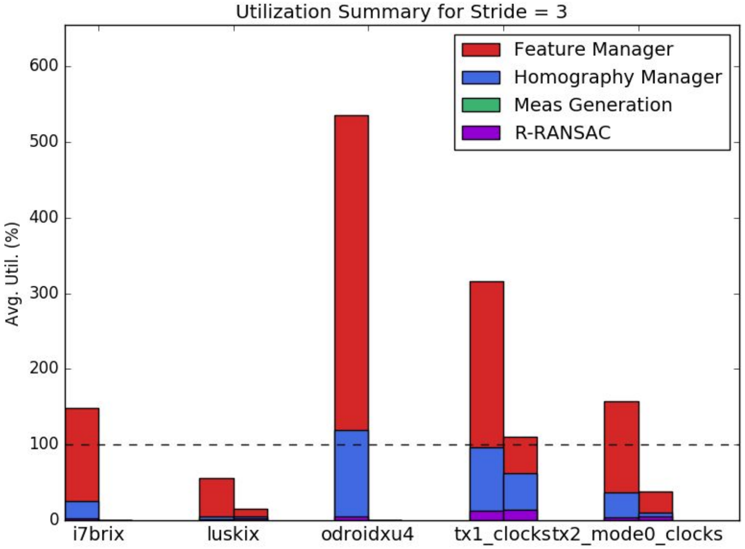
- Specifications
  - GPU: 256 Pascal CUDA Cores
  - CPU: Dual Denver + Quad ARM A57
  - 8GB RAM
  - 32GB eMMC
- Attach to UAV with ConnectTech Inc Orbitty carrier board



HW Platform Runtime Comparison

- TX2 with GPU can handle 10Hz and 15Hz with ~50% CPU/GPU utilization





C-UAS

Summary

- Future holds exciting opportunities for small UAS
- Many challenges with vision based flight
  - Long term GPS degraded navigation
  - Robust object tracking