

# Real Time, Onboard Landing Site Evaluation for Autonomous Drones

## PhD Thesis Proposal

Joshua Springer

Reykjavík University

March 2022



# Presentation Structure

## (1) Introduction

- Problem description and motivation
- State of the Art

## (2) Completed/ongoing projects

- Initial proof of concept attempt
  - Continuation of master thesis (tested in simulation)
- Fiducial marker modifications
- Proof of concept

## (3) Research Plan

- Methods
- Challenges and risk analysis



# Introduction



# Problem Description and Motivation

- Much of basic drone flight has been **automated**.



# Problem Description and Motivation

- Much of basic drone flight has been **automated**.
  - Takeoff
  - Waypoint-to-waypoint-flight
  - Track/orbit objects,  
take pictures, etc.



# Problem Description and Motivation

- Much of basic drone flight has been **automated**.
  - Takeoff
  - Waypoint-to-waypoint-flight
  - Track/orbit objects,  
take pictures, etc.
- Landing is still largely **manual**.



# Problem Description and Motivation

- Much of basic drone flight has been **automated**.
  - Takeoff
  - Waypoint-to-waypoint-flight
  - Track/orbit objects,  
take pictures, etc.
- Landing is still largely **manual**.
  - No continuous, autonomous mission cycles
  - Primitive, semi-autonomous methods are common  
(still require human operator)
  - Hand-catching is common



# Problem Description and Motivation

- Much of basic drone flight has been **automated**.
  - Takeoff
  - Waypoint-to-waypoint-flight
  - Track/orbit objects,  
take pictures, etc.
- Landing is still largely **manual**.
  - No continuous, autonomous mission cycles
  - Primitive, semi-autonomous methods are common  
(still require human operator)
  - Hand-catching is common



“Human-assisted landing”



# Research Questions



# Research Questions

- How can a drone autonomously land?



# Research Questions

- How can a drone autonomously land?
- What data do autonomous drone landing methods need?



# Research Questions

- How can a drone autonomously land?
- What data do autonomous drone landing methods need?
- How can those methods execute in real time onboard a drone?



# State of the Art

- GPS-based landing



# State of the Art

- GPS-based landing
  - RTK



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers
  - IR beacons



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers
  - IR beacons
- Terrain analysis
  - Optical flow



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers
  - IR beacons
- Terrain analysis
  - Optical flow (requires motion)



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers
  - IR beacons
- Terrain analysis
  - Optical flow (requires motion)
  - RGBD, LIDAR



# State of the Art

- GPS-based landing
  - RTK
- Known landing locations:
  - Visual matching
  - Visual markers
  - IR beacons
- Terrain analysis
  - Optical flow (requires motion)
  - RGBD, LIDAR (slow, offload processing)



# Completed and Ongoing Projects



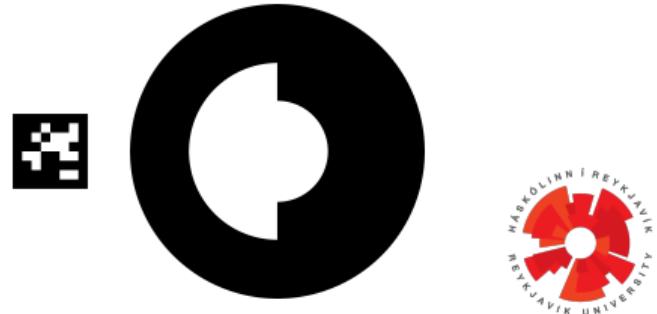
# Test Hexacopters

- Two Tarot 680 hexacopters
- For real-world proof of concept of master thesis simulations.



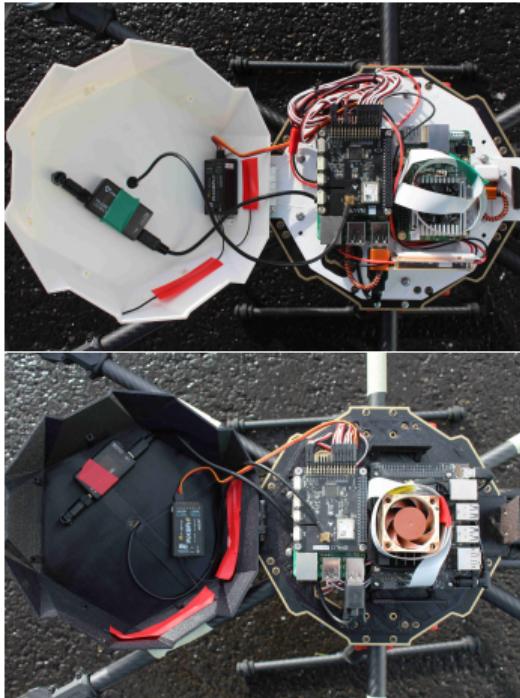
# Test Hexacopters

- Two Tarot 680 hexacopters
- For real-world proof of concept of master thesis simulations.



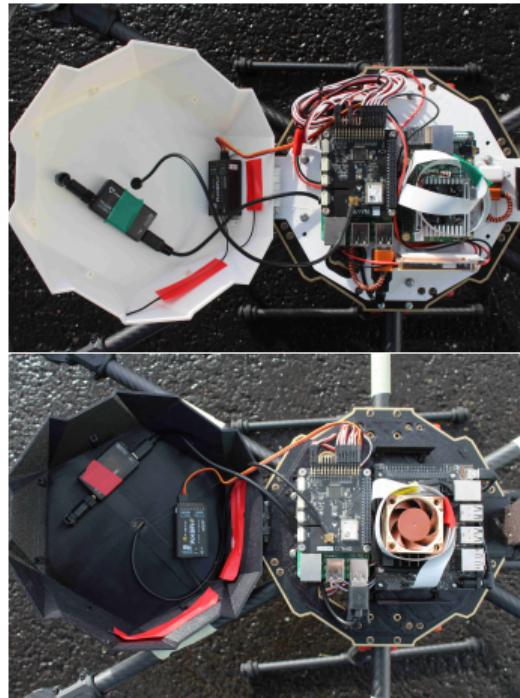
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo



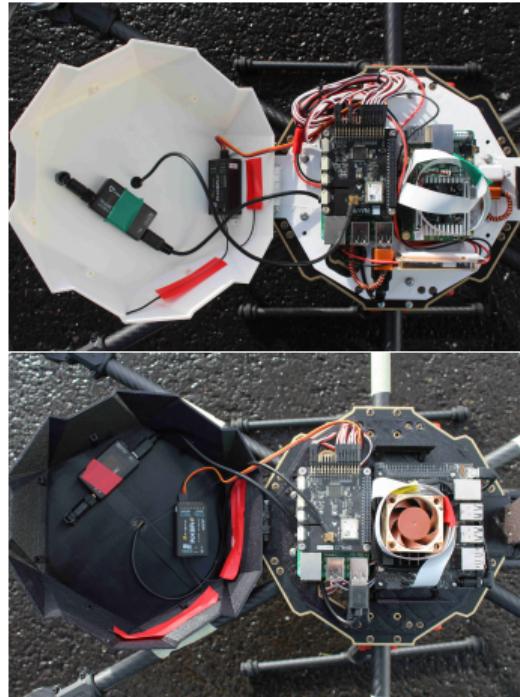
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):



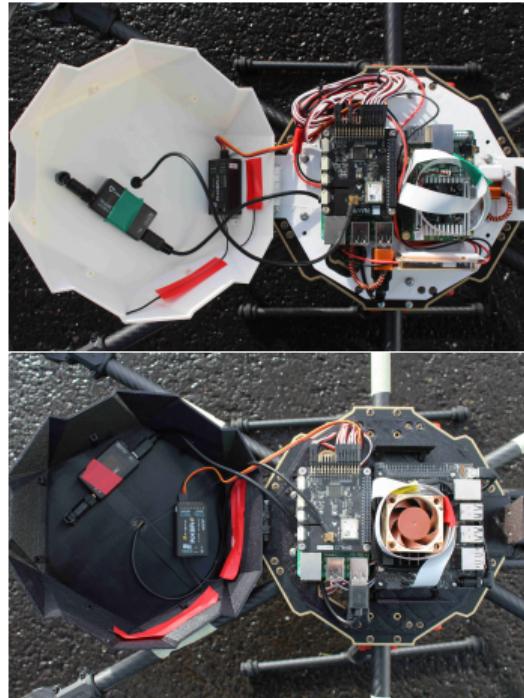
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)



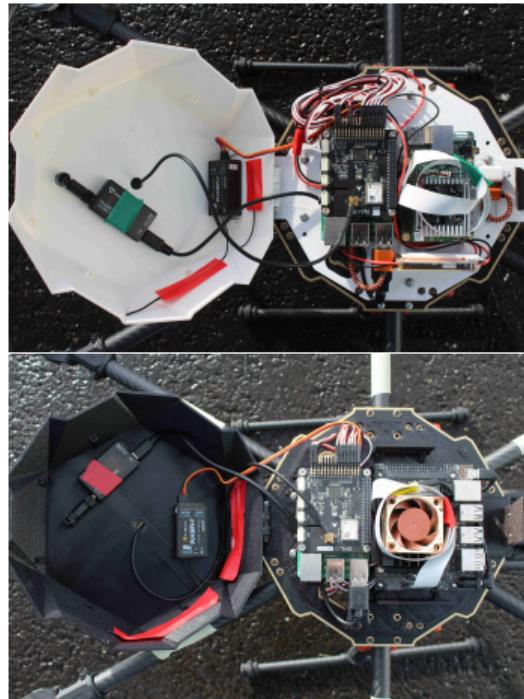
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)



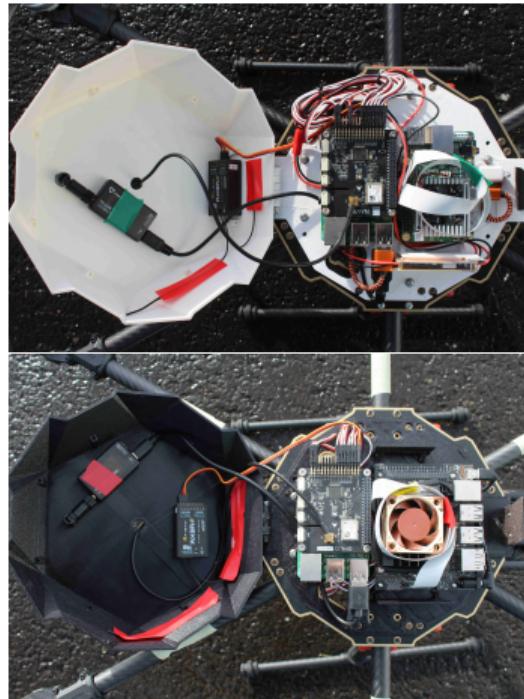
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)
- Gimbaled camera modules



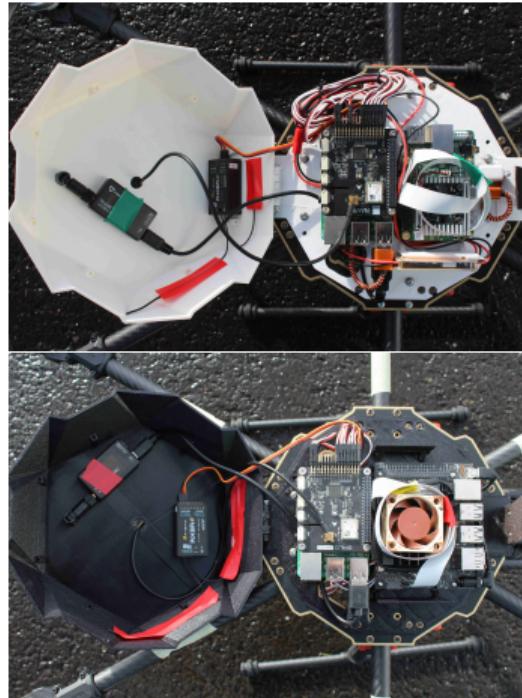
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)
- Gimbaled camera modules
- 433 MHz telemetry



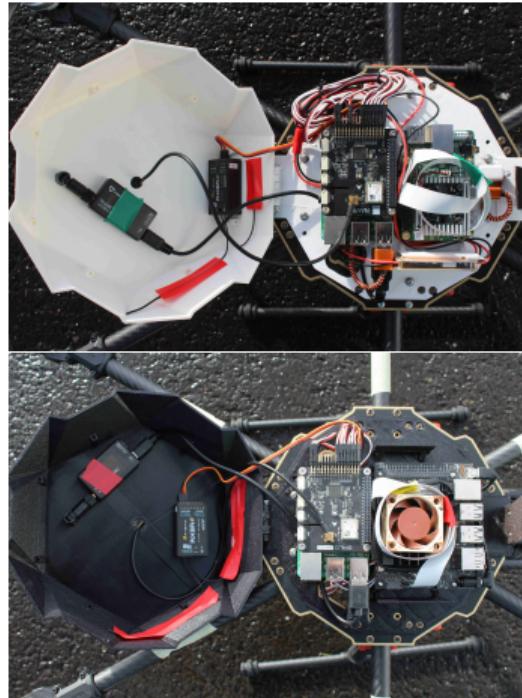
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)
- Gimbaled camera modules
- 433 MHz telemetry
- 2.4 GHz R/C control



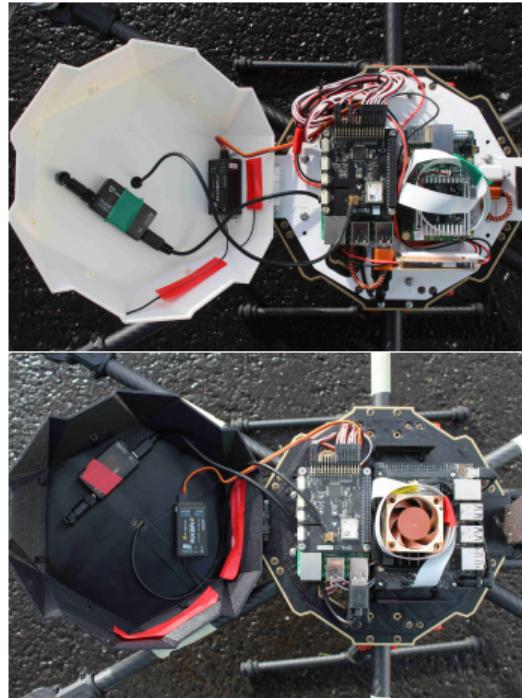
# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)
- Gimbaled camera modules
- 433 MHz telemetry
- 2.4 GHz R/C control
- Tested Autopilot Softwares
  - ArduPilot



# Test Hexacopter Components

- Navio2 + RPi 3 autopilot combo
- Companion boards (for heavy onboard processing):
  - Google Coral (embedded TPU)
  - Jetson Nano (embedded GPU)
- Gimbaled camera modules
- 433 MHz telemetry
- 2.4 GHz R/C control
- Tested Autopilot Softwares
  - ArduPilot
  - PX4 (not technically supported)



# Test Hexacopters' Performance

- Stable (manual) flight performance



# Test Hexacopters' Performance

- Stable (manual) flight performance
- ~20 min flying time



# Test Hexacopters' Performance

- Stable (manual) flight performance
- ~20 min flying time
- Successful marker tracking



# Test Hexacopters' Performance

- Stable (manual) flight performance
- ~20 min flying time
- Successful marker tracking
- Errors during approach
  - Monocular pose estimation ambiguity



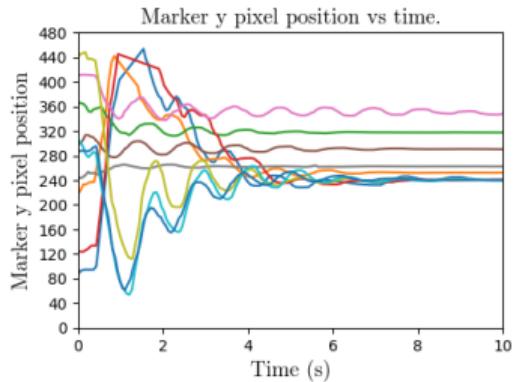
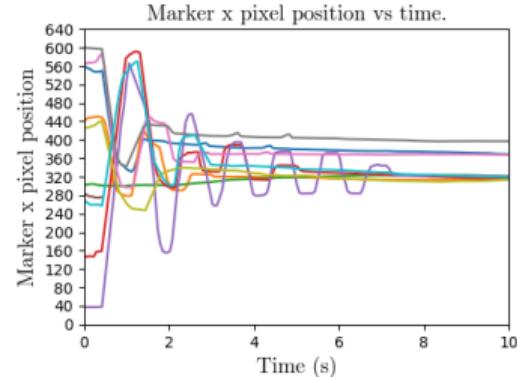
# Test Hexacopters' Performance

- Stable (manual) flight performance
- ~20 min flying time
- Successful marker tracking
- Errors during approach
  - Monocular pose estimation ambiguity
  - GPS inaccuracy



# Test Hexacopters' Performance

- Stable (manual) flight performance
- ~20 min flying time
- Successful marker tracking
- Errors during approach
  - Monocular pose estimation ambiguity
  - GPS inaccuracy
- No successful autonomous landing  
(but almost)



Meanwhile...



# Heavy Lift IR Drone

- Project with Christopher Hamilton (geologist, University of Arizona) and Baldur Björnsson



# Heavy Lift IR Drone

- Project with Christopher Hamilton (geologist, University of Arizona) and Baldur Björnsson
- 1.3 m span, 25 kg lift



# Heavy Lift IR Drone

- Project with Christopher Hamilton (geologist, University of Arizona) and Baldur Björnsson
- 1.3 m span, 25 kg lift
- FLIR camera



# Heavy Lift IR Drone

- Project with Christopher Hamilton (geologist, University of Arizona) and Baldur Björnsson
- 1.3 m span, 25 kg lift
- FLIR camera
- Surveyed lava field at Fagradalsfjall



# Heavy Lift IR Drone

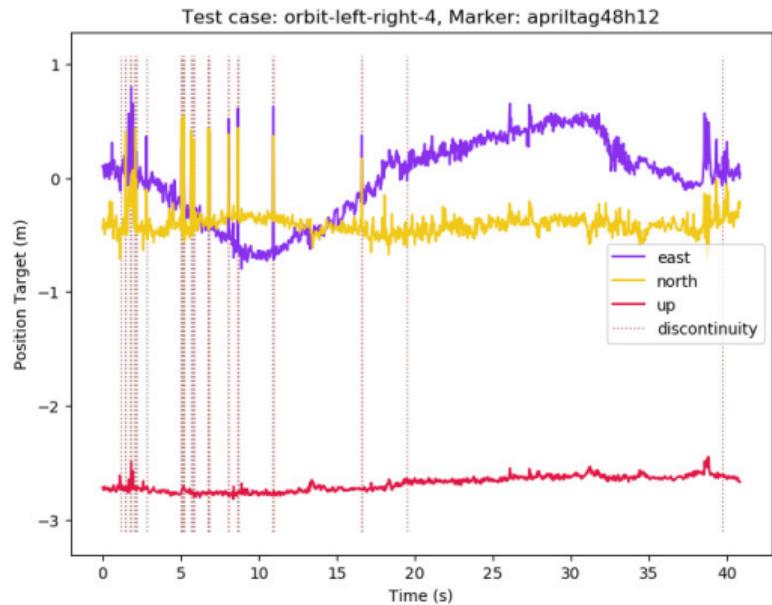
- Project with Christopher Hamilton (geologist, University of Arizona) and Baldur Björnsson
- 1.3 m span, 25 kg lift
- FLIR camera
- Surveyed lava field at Fagradalsfjall
- Featured on BBC Click



# Fiducial System Modifications

Necessary properties:

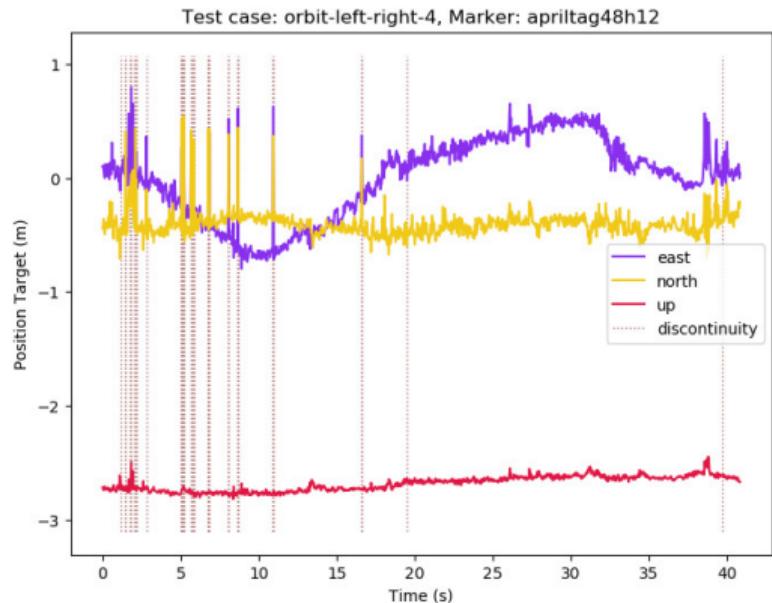
- Mitigates orientation ambiguity



# Fiducial System Modifications

Necessary properties:

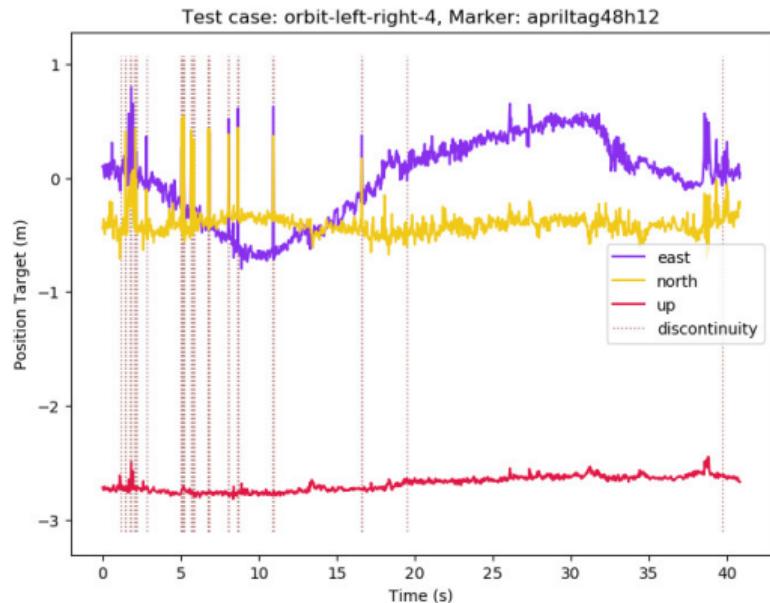
- Mitigates orientation ambiguity
- Detectable at long- and short-range



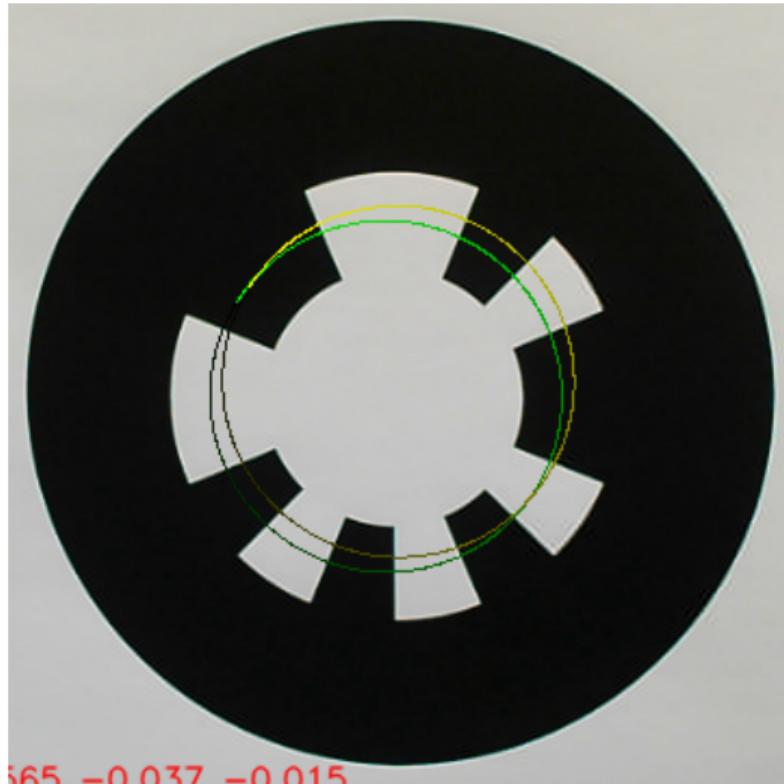
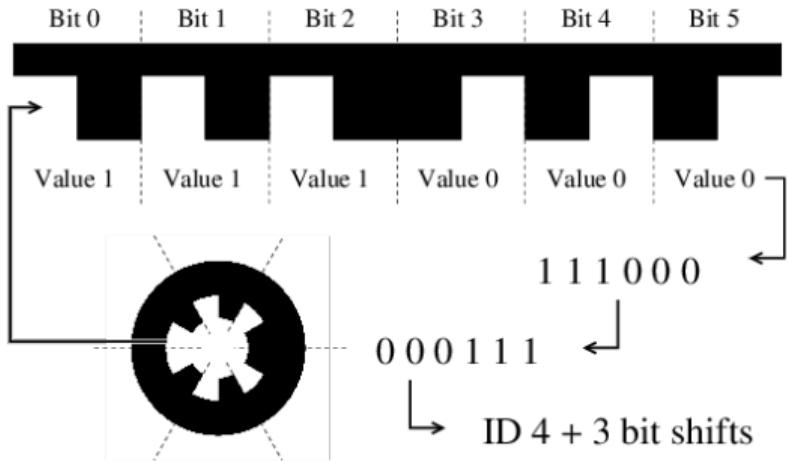
# Fiducial System Modifications

Necessary properties:

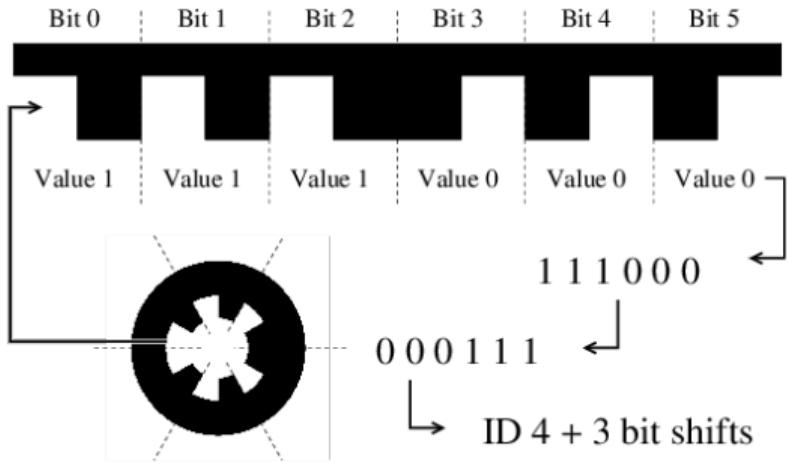
- Mitigates orientation ambiguity
- Detectable at long- and short-range
- Runs on embedded hardware



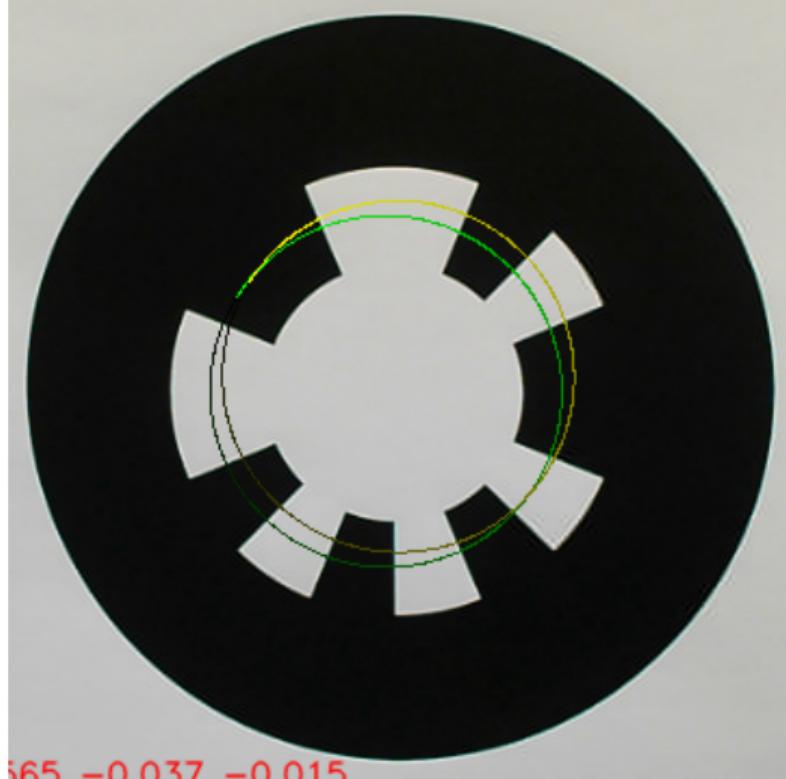
# WhyCode Orig(inal)



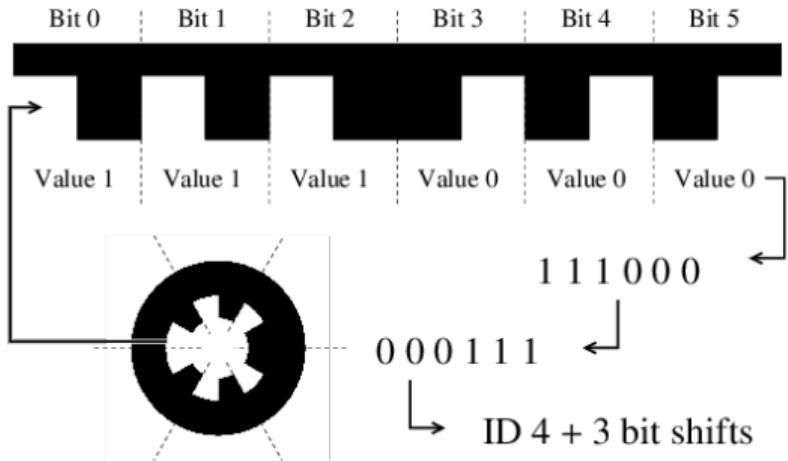
# WhyCode Orig(inal)



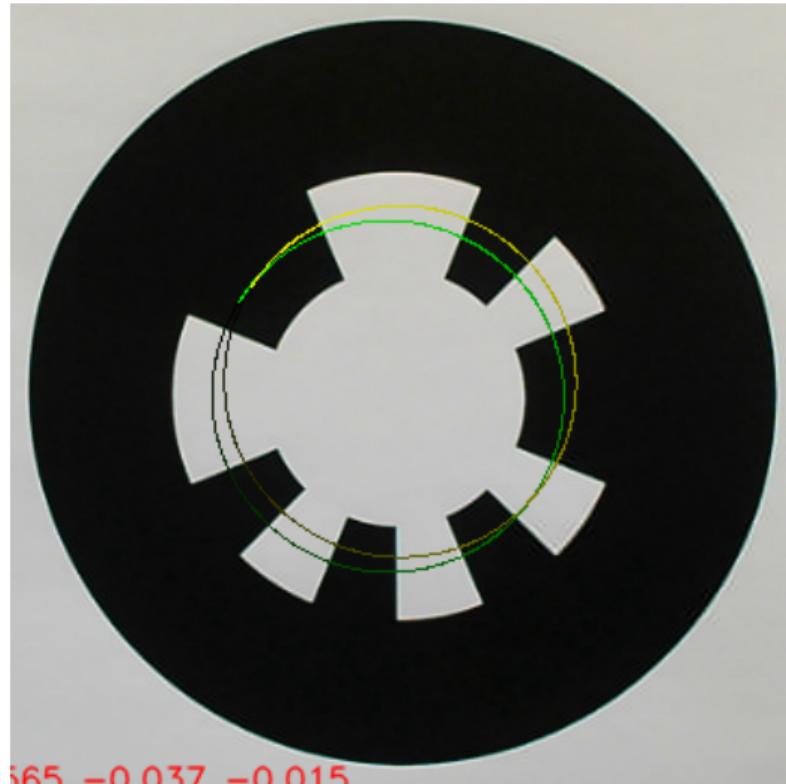
- Semi-axes → 2 possible orientations



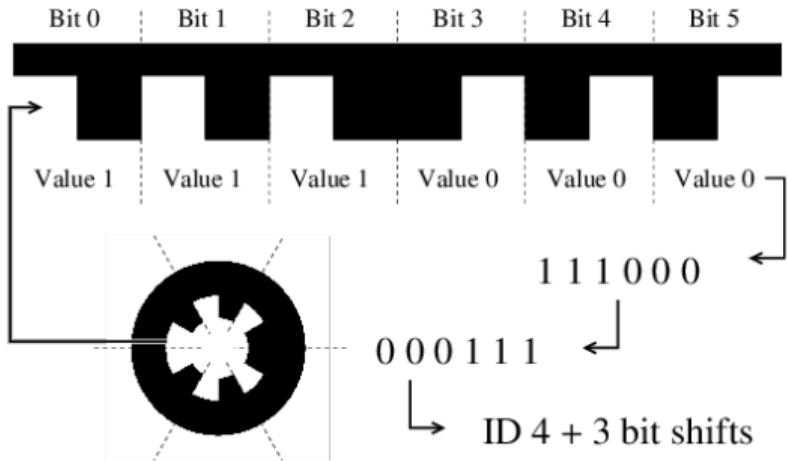
# WhyCode Orig(inal)



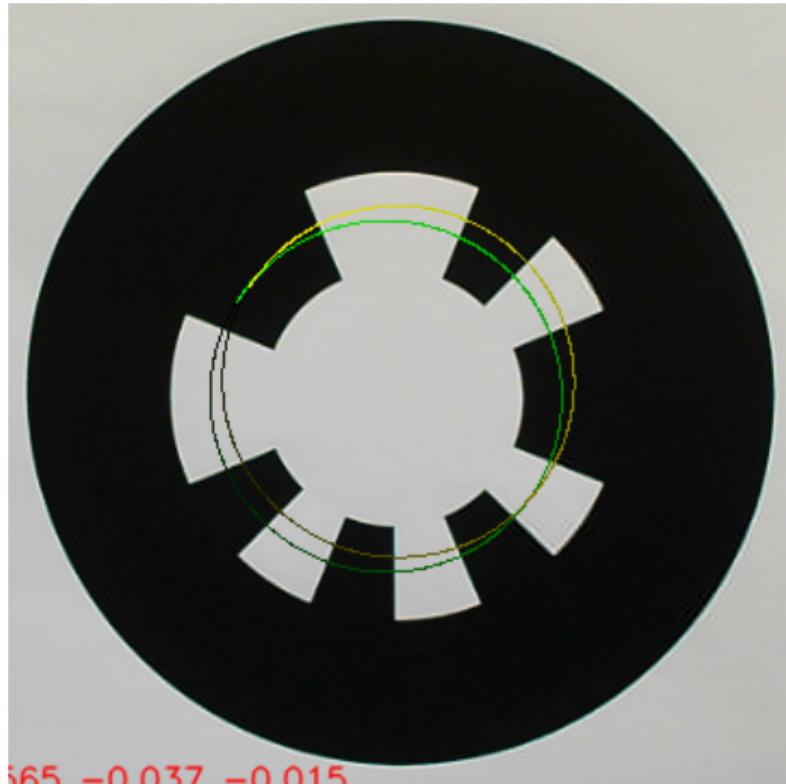
- Semi-axes → 2 possible orientations
- Better centered → correct



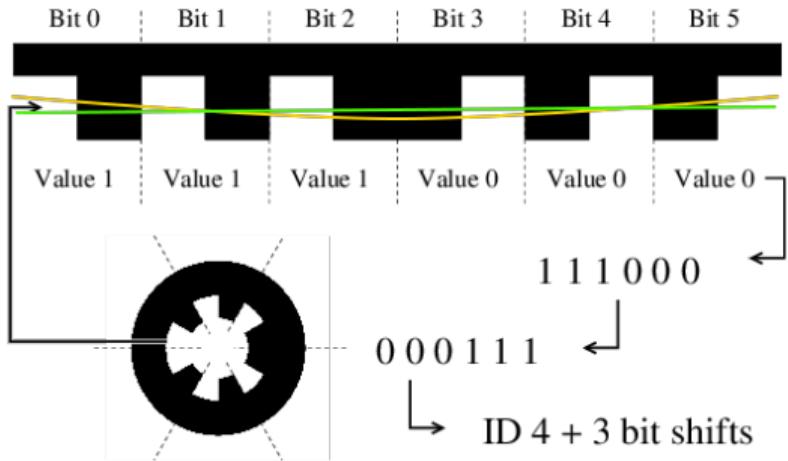
# WhyCode Orig(inal)



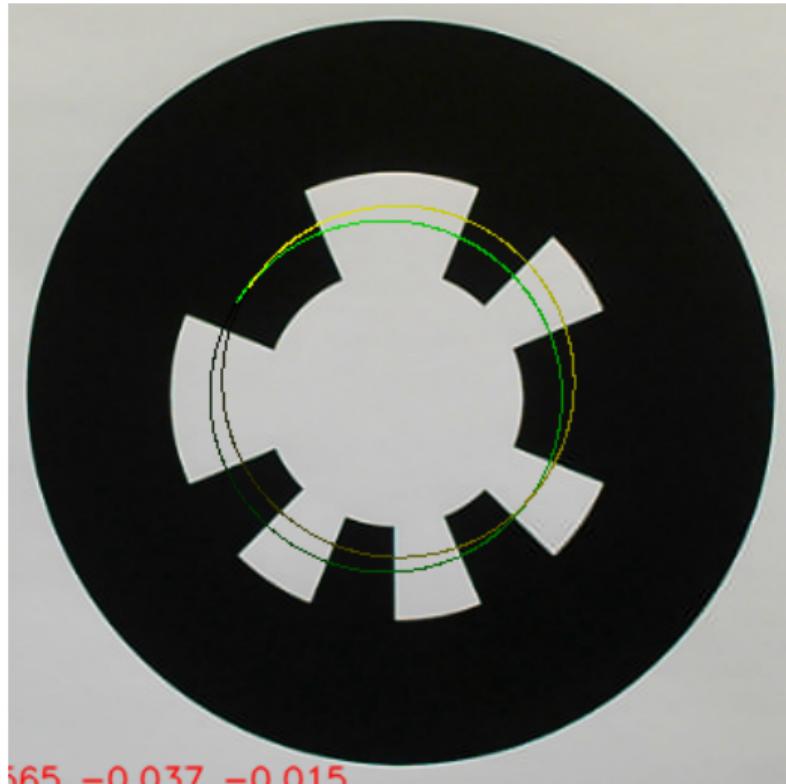
- Semi-axes → 2 possible orientations
- Better centered → correct
- Arclength of intersections with ID “teeth”



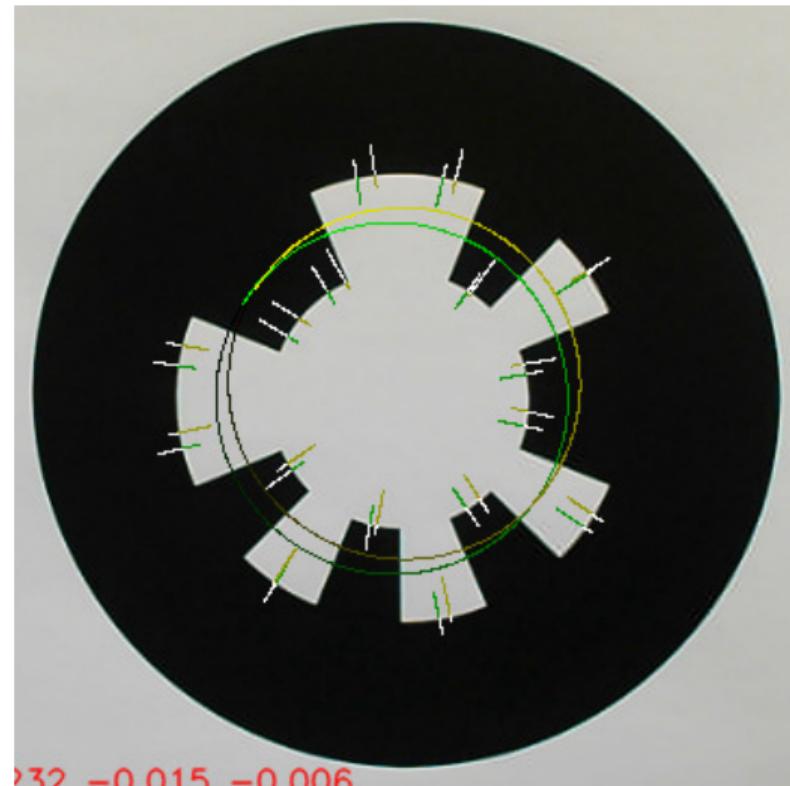
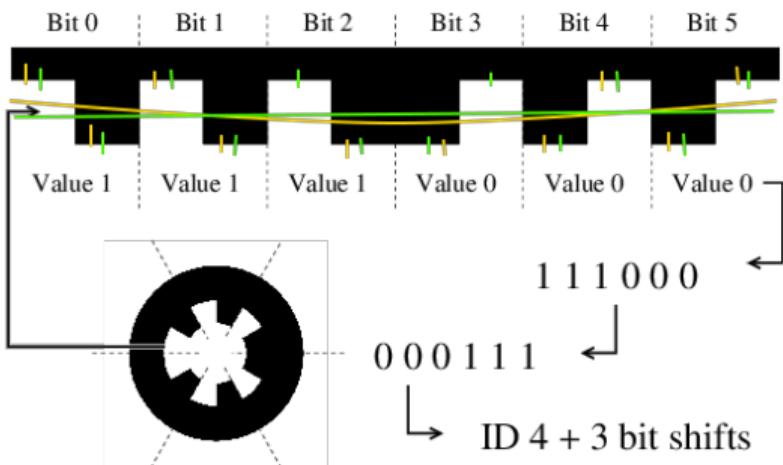
# WhyCode Orig(inal)



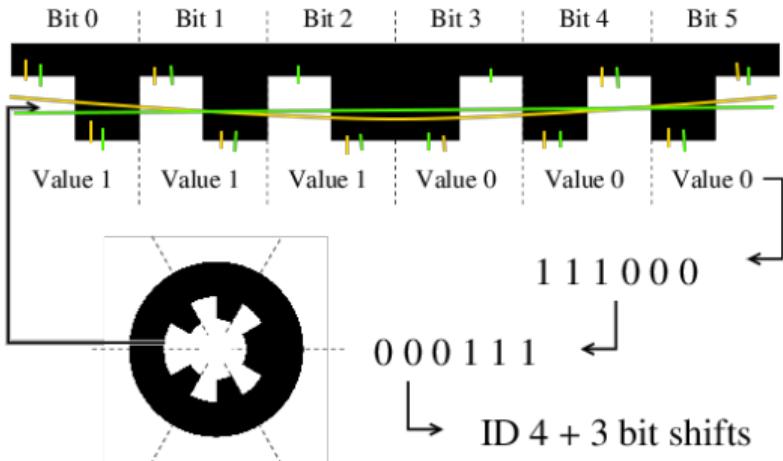
- Semi-axes → 2 possible orientations
- Better centered → correct
- Arclength of intersections with ID “teeth”



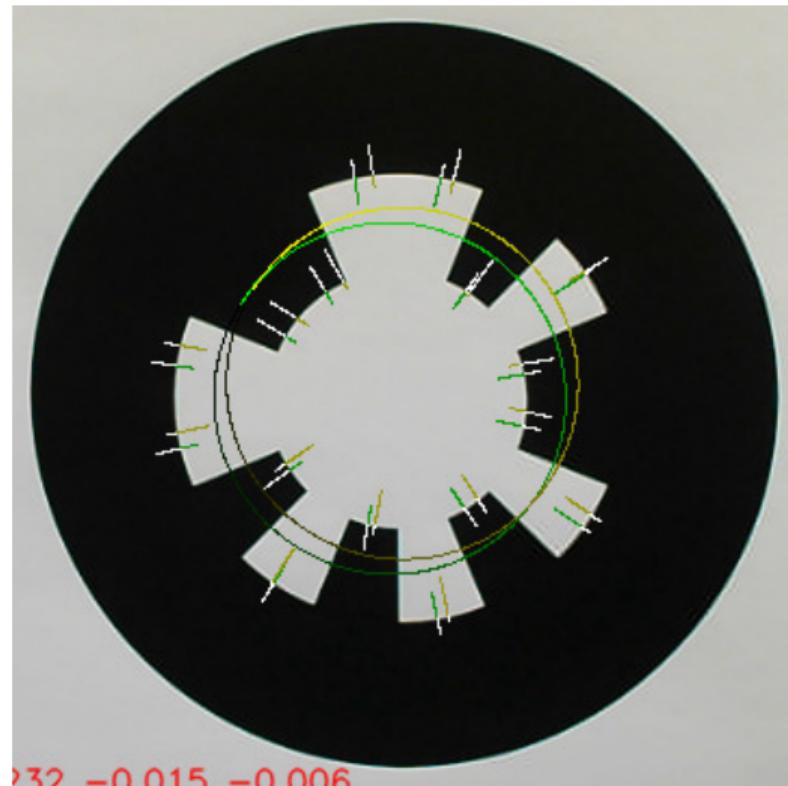
# Fiducial System Modifications: “WhyCode Ellipse”



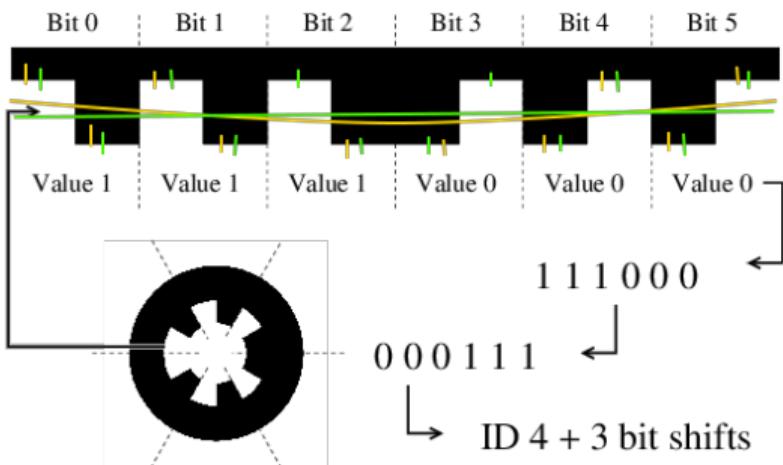
# Fiducial System Modifications: “WhyCode Ellipse”



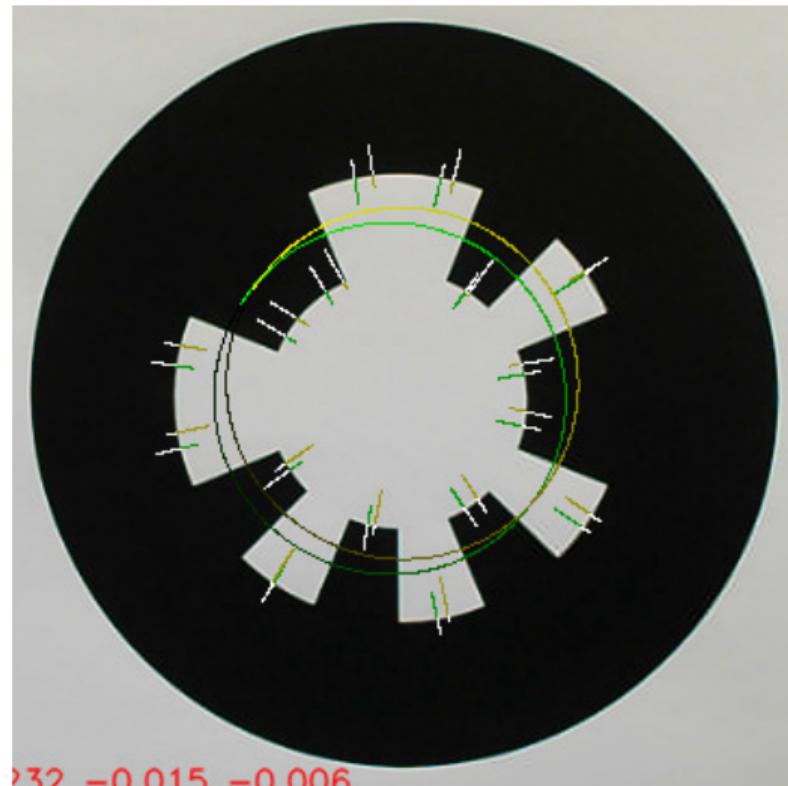
- Sample ID with original method.



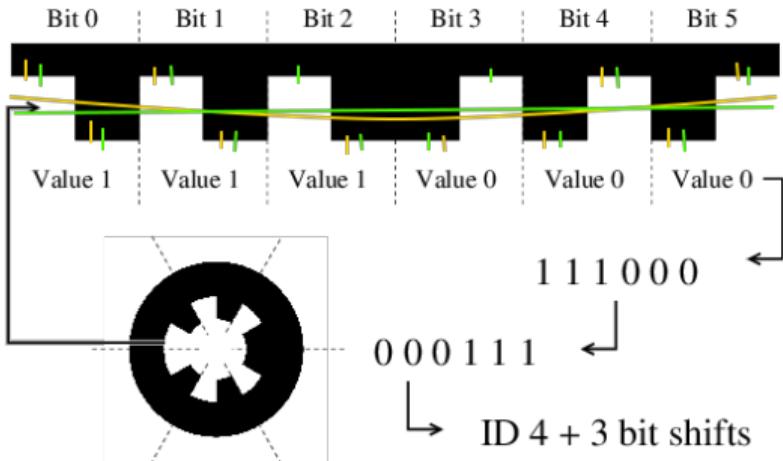
# Fiducial System Modifications: “WhyCode Ellipse”



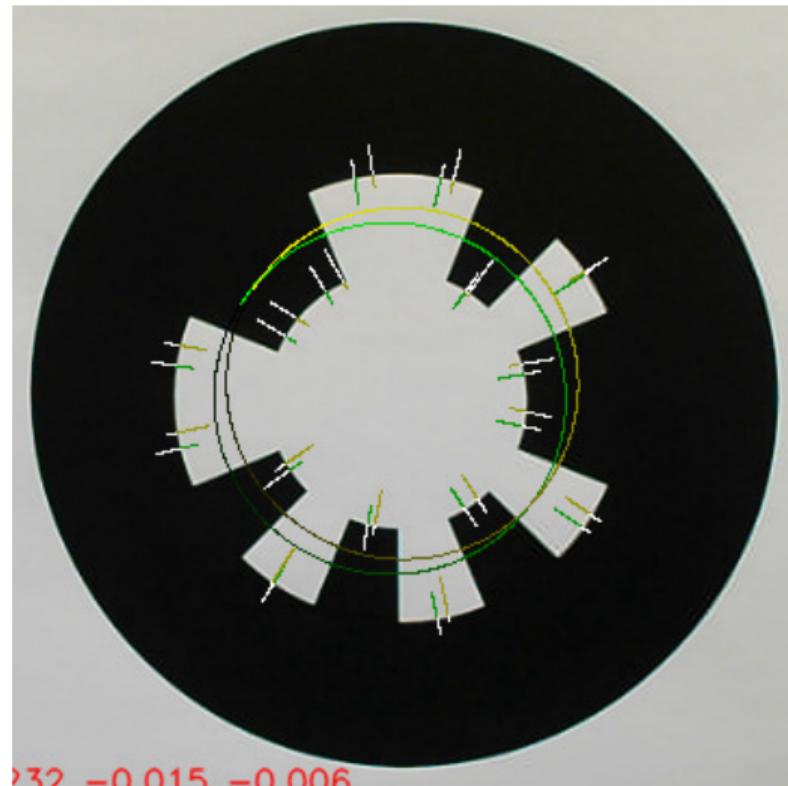
- Sample ID with original method.
- Add: radial sampling on tooth edges.



# Fiducial System Modifications: “WhyCode Ellipse”

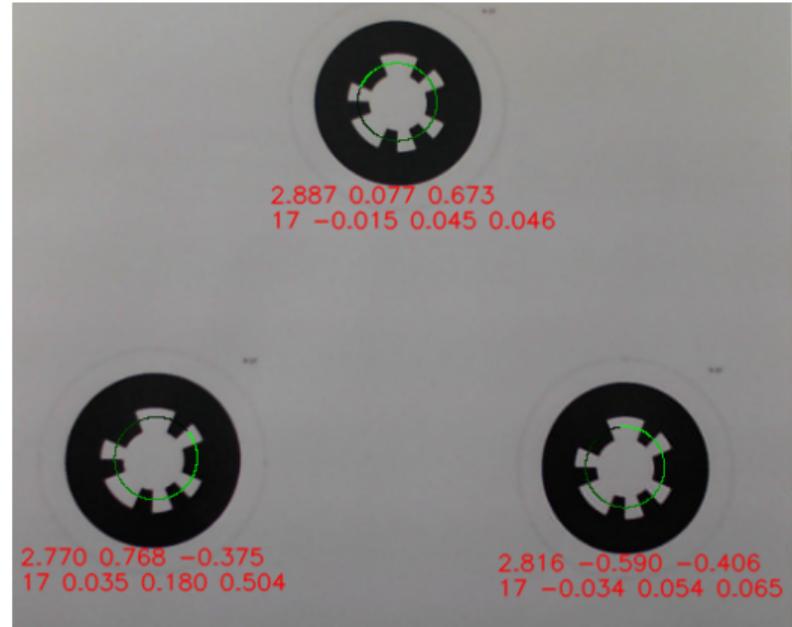


- Sample ID with original method.
- Add: radial sampling on tooth edges.
- Minimize variance on extra sample lines.



# Fiducial System Modifications: “WhyCode Multi”

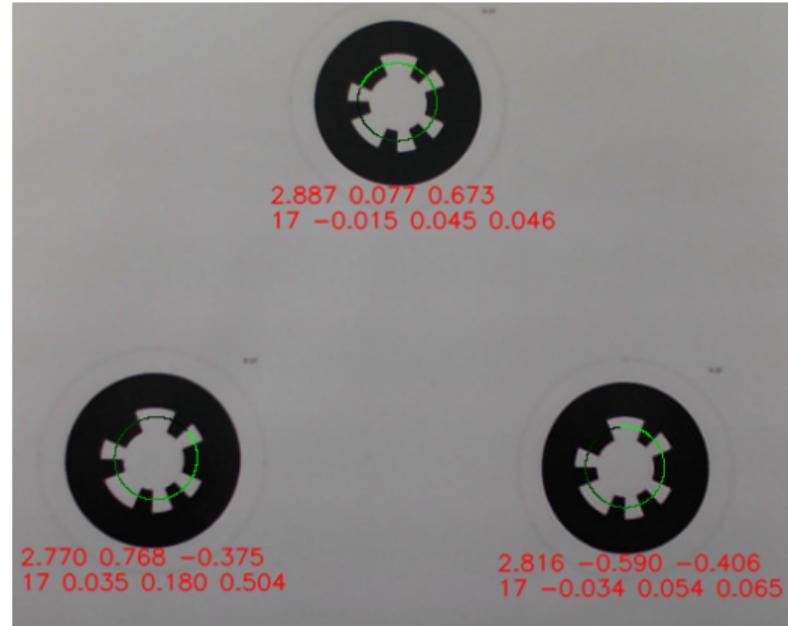
## Approach 2: Coplanar marker arrangements



# Fiducial System Modifications: “WhyCode Multi”

## Approach 2: Coplanar marker arrangements

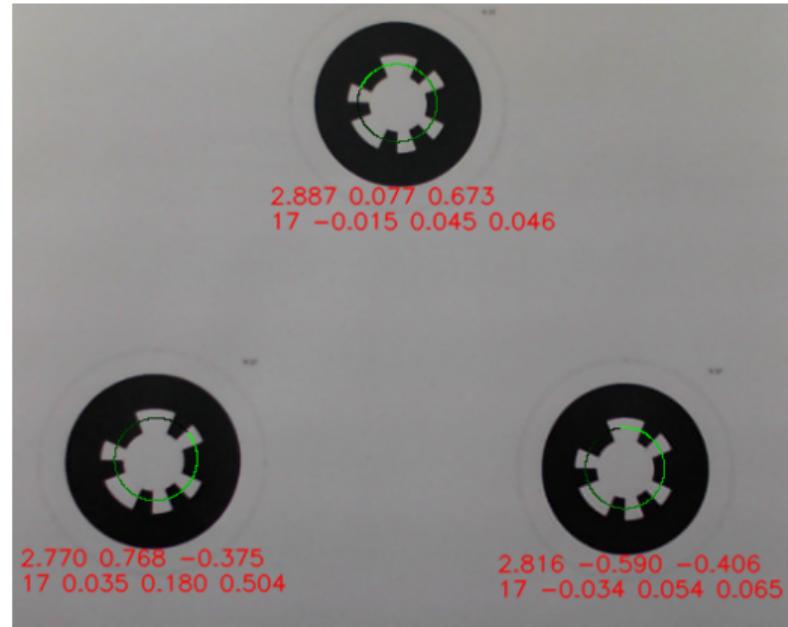
- Ignore individual marker orientations



# Fiducial System Modifications: “WhyCode Multi”

## Approach 2: Coplanar marker arrangements

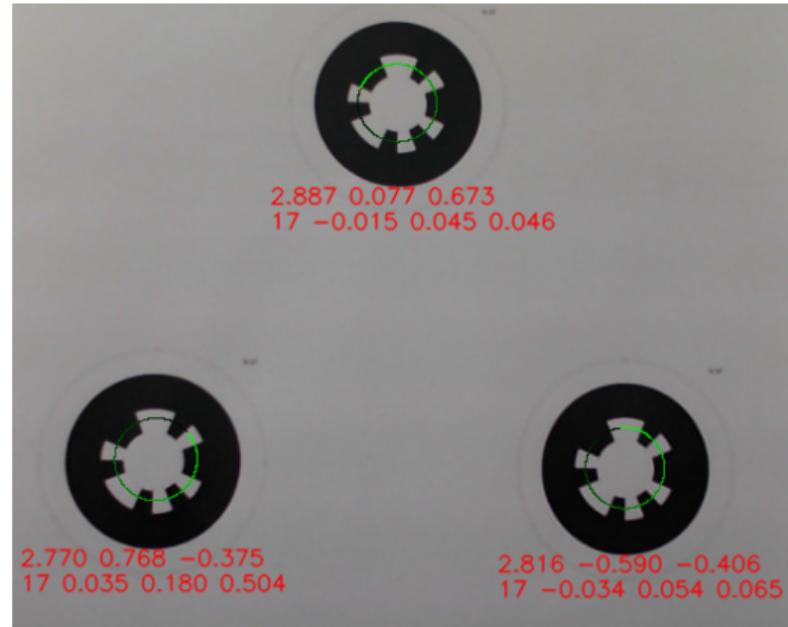
- Ignore individual marker orientations
- Calculate normal vector to the plane connecting the markers.



# Fiducial System Modifications: “WhyCode Multi”

## Approach 2: Coplanar marker arrangements

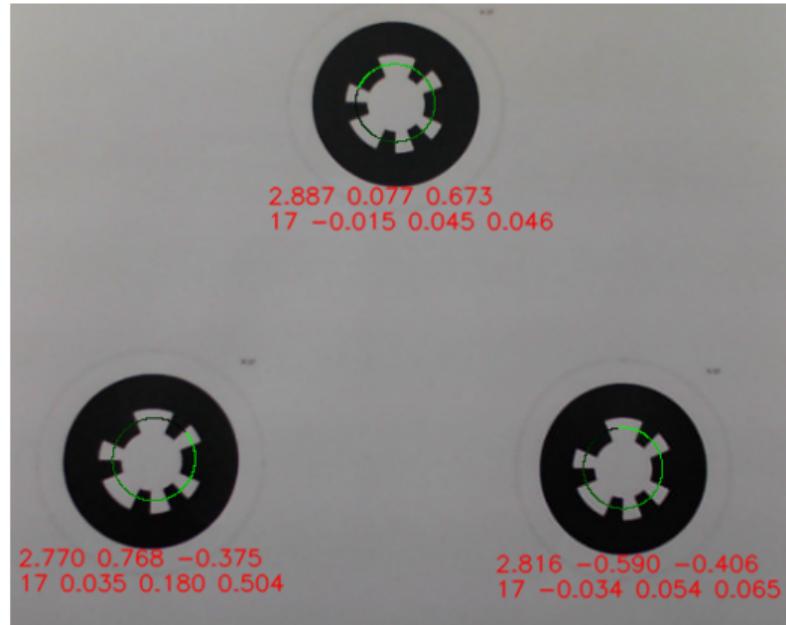
- Ignore individual marker orientations
- Calculate normal vector to the plane connecting the markers.
- Extract pitch and roll from the normal vector.



# Fiducial System Modifications: “WhyCode Multi”

## Approach 2: Coplanar marker arrangements

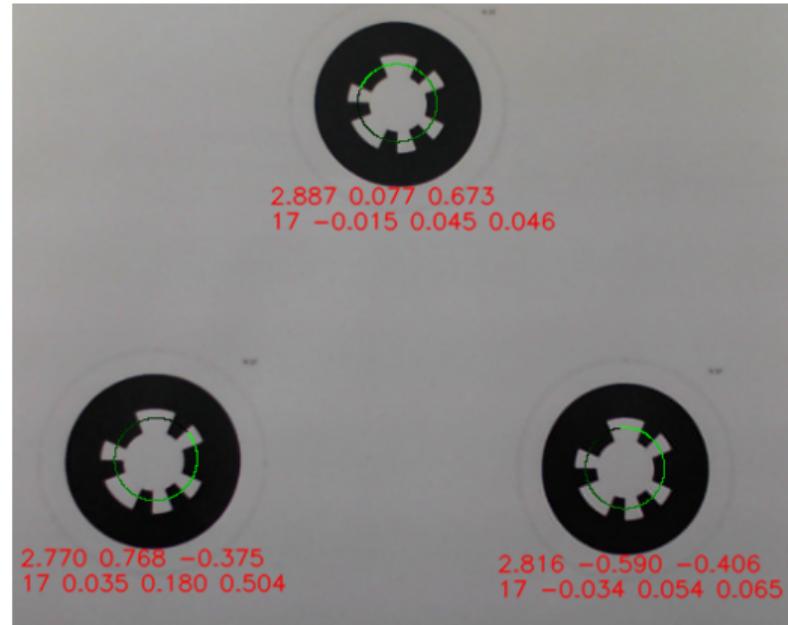
- Ignore individual marker orientations
- Calculate normal vector to the plane connecting the markers.
- Extract pitch and roll from the normal vector.
- Extract yaw from the marker IDs.



# Fiducial System Modifications: “WhyCode Multi”

## Approach 2: Coplanar marker arrangements

- Ignore individual marker orientations
- Calculate normal vector to the plane connecting the markers.
- Extract pitch and roll from the normal vector.
- Extract yaw from the marker IDs.
- Takes advantage of WhyCode’s efficiency.



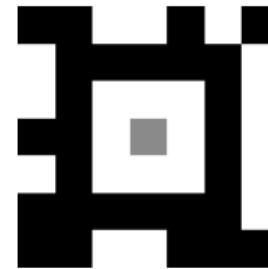
# Fiducial System Modifications: April Tag

April Tag: less orientation ambiguity, but less computationally efficient.

April Tag 48h12: more sophisticated, “recursive.”

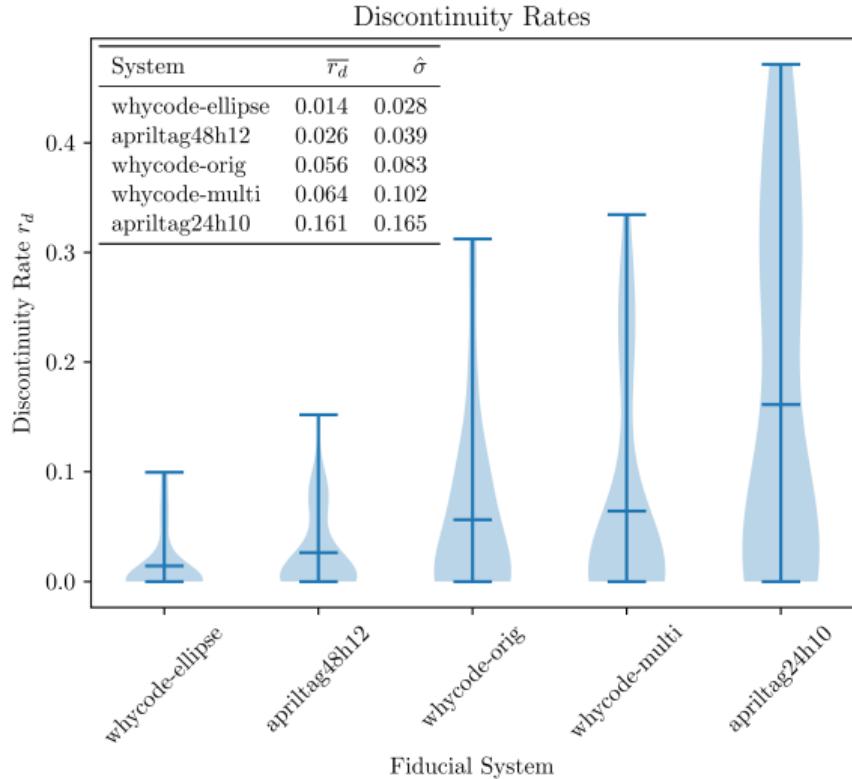


April Tag Custom 24h10: “recursive,” smaller definition



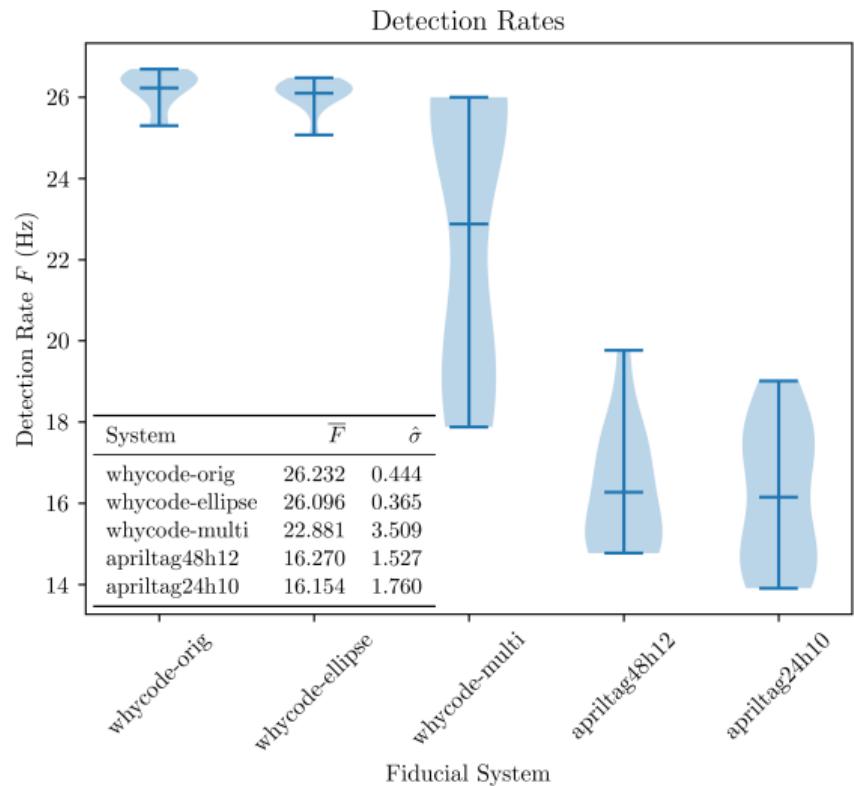
# Performance Analysis: Discontinuity Rates

- Orientation ambiguity → discontinuities.
- Discontinuity rate  $\bar{r}_d$  is the number of discontinuities per detection.
- Lower is better.



# Performance Analysis: Detection Rates

- Detection rate  $\bar{F}$  is the number of detections per second.
- Tested on Raspberry Pi 4.
- Higher is better.



# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark



(Banana for scale.)

# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather



(Banana for scale.)

# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather
  - Stable out-of-the-box autonomous flight



(Banana for scale.)

# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather
  - Stable out-of-the-box autonomous flight
  - Doesn't require GPS (uses other sensors)



(Banana for scale.)

# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather
  - Stable out-of-the-box autonomous flight
  - Doesn't require GPS (uses other sensors)
- Requires DJI Mobile SDK, Custom Android App, and **lots** of workarounds.



(Banana for scale.)

# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather
  - Stable out-of-the-box autonomous flight
  - Doesn't require GPS (uses other sensors)
- Requires DJI Mobile SDK, Custom Android App, and **lots** of workarounds.
- Video frames are offloaded (via WiFi) to Raspberry Pi 4 for processing



(Banana for scale.)

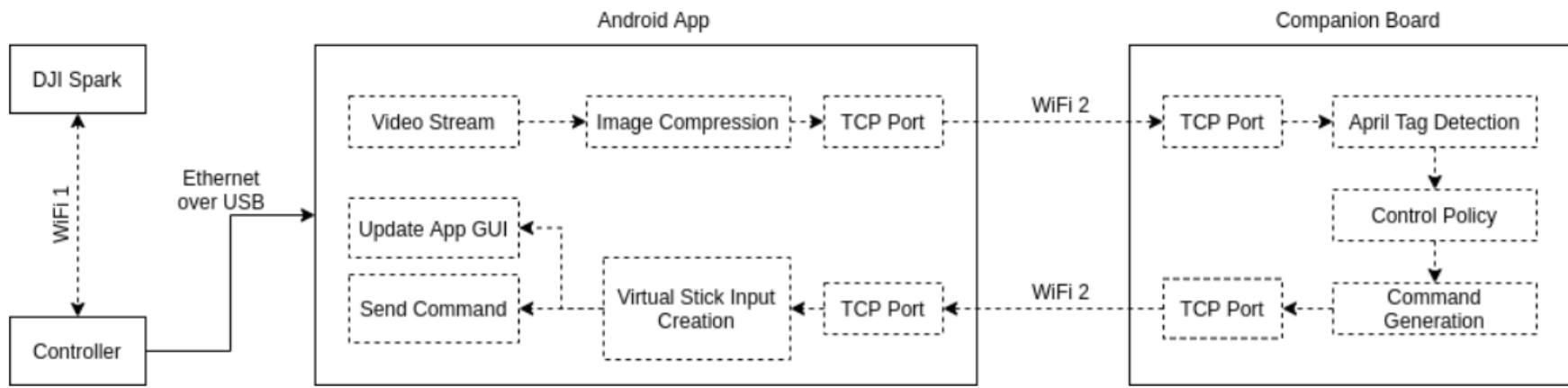
# Autonomous Landing Proof of Concept

- Indoor experiments with DJI Spark
  - Reduces logistical considerations: transportation, weather
  - Stable out-of-the-box autonomous flight
  - Doesn't require GPS (uses other sensors)
- Requires DJI Mobile SDK, Custom Android App, and **lots** of workarounds.
- Video frames are offloaded (via WiFi) to Raspberry Pi 4 for processing
- Limiting factor: pre-transmission image compression on tablet (6-7 Hz)

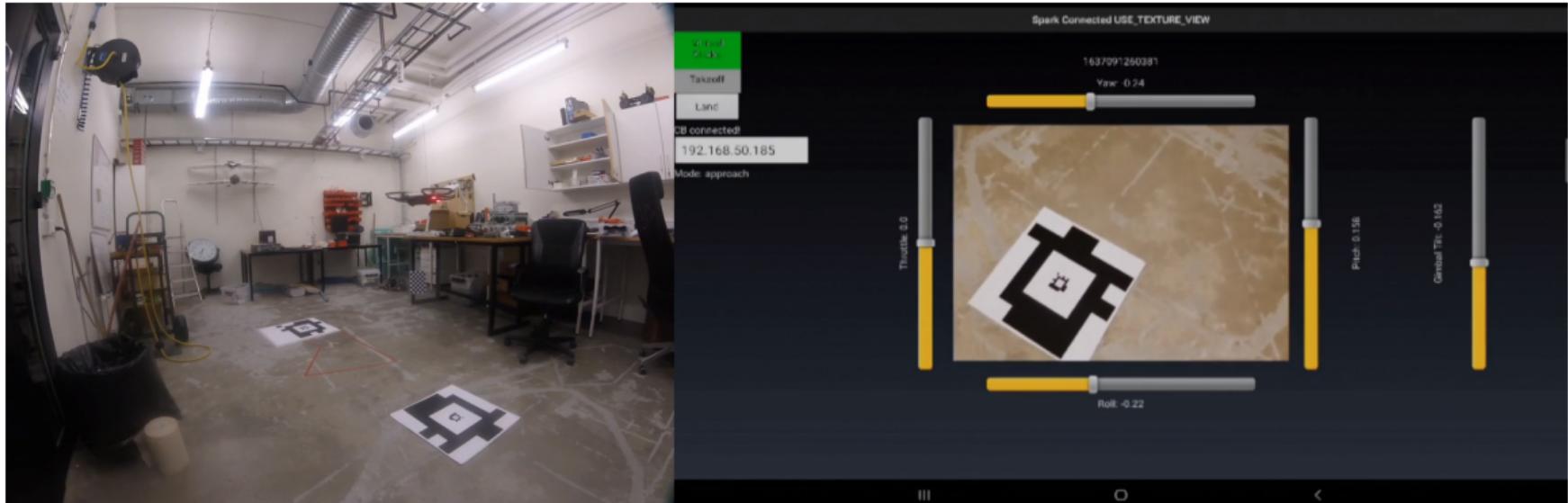


(Banana for scale.)

# Autonomous Landing Proof of Concept: System Architecture



# Demo with worst-performing April Tag 24h10



Works with every system except WhyCode Multi.

# Publications

- Submitted: Evaluation of April Tag and WhyCode Fiducial Systems for Autonomous Precision Drone Landing with a Gimbal-Mounted Camera
- In Progress: results from autonomous landing proof of concept



# Research Plan



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)
- Focus on real time performance



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)
- Focus on real time performance
  - Minimize computational requirements



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)
- Focus on real time performance
  - Minimize computational requirements
  - Target specific hardware platforms



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)
- Focus on real time performance
  - Minimize computational requirements
  - Target specific hardware platforms
- Overall structure:
  - Input: sensor data
  - Process (quickly): ??
  - Output: safe landing sites (e.g. heat map)



# Overview: Unstructured Autonomous Landing

- Focus on terrain analysis
  - Topographical analysis
  - Semantic segmentation
    - terrain type classification: (snow, ice, water, grass, rock, etc.)
    - classify according to predicted safety: (safe, questionable, unsafe, etc.)
- Focus on real time performance
  - Minimize computational requirements
  - Target specific hardware platforms
- Overall structure:
  - Input: sensor data
  - Process (quickly): ??
  - Output: safe landing sites (e.g. heat map) → flight control commands



# Sensor Data



# Data Set Generation

AirSim: realistic simulator

- Automatic generation of large data sets

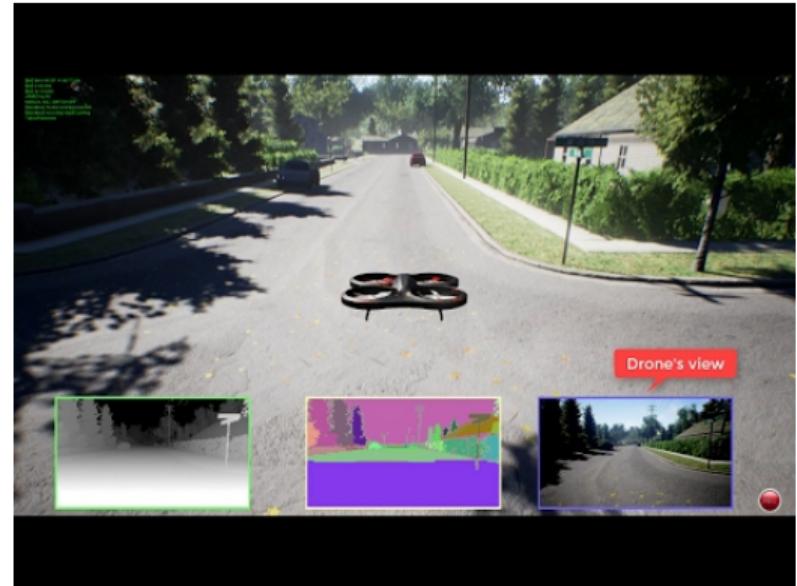


Image source



# Data Set Generation

AirSim: realistic simulator

- Automatic generation of large data sets
- Synthetic sensor data
  - RGBD
  - LIDAR
  - RADAR
  - IMU
  - Specify parameters

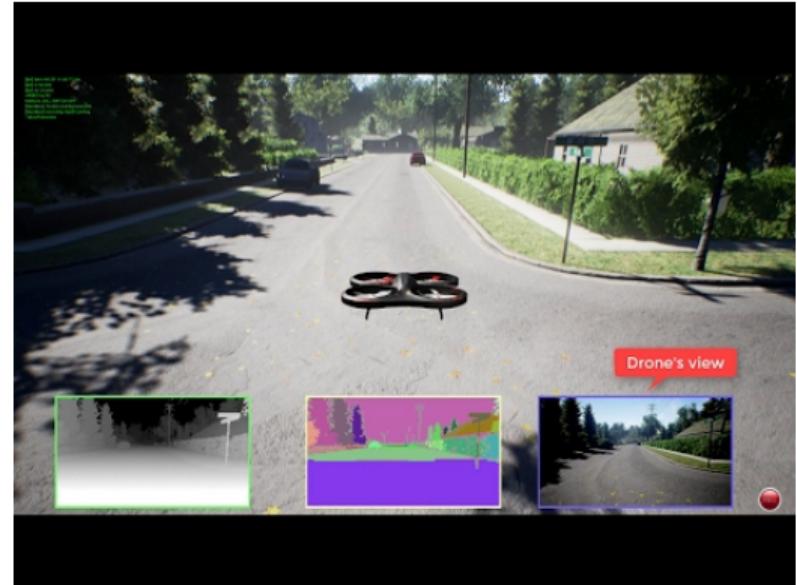


Image source



# Data Set Generation

AirSim: realistic simulator

- Automatic generation of large data sets
- Synthetic sensor data
  - RGBD
  - LIDAR
  - RADAR
  - IMU
  - Specify parameters
- Segmentation masks

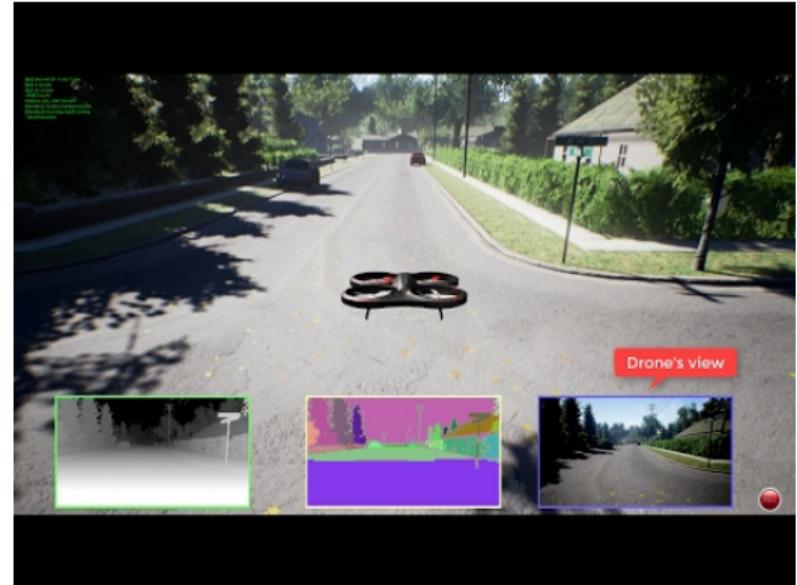


Image source



# Data Set Generation

AirSim: realistic simulator

- Automatic generation of large data sets
- Synthetic sensor data
  - RGBD
  - LIDAR
  - RADAR
  - IMU
  - Specify parameters
- Segmentation masks
- Labeling method can be slow, hand-tuned

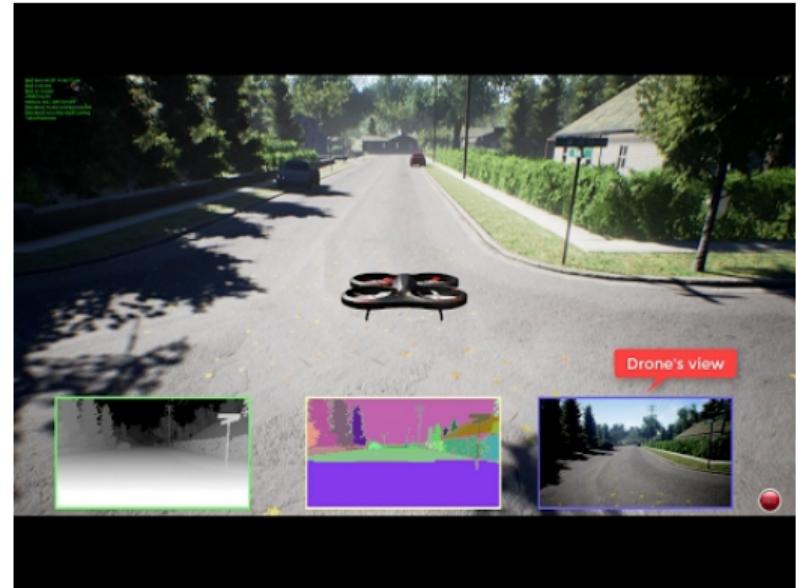


Image source



# Data Set Generation

AirSim: realistic simulator

- Automatic generation of large data sets
- Synthetic sensor data
  - RGBD
  - LIDAR
  - RADAR
  - IMU
  - Specify parameters
- Segmentation masks
- Labeling method can be slow, hand-tuned
- Real world data for validation

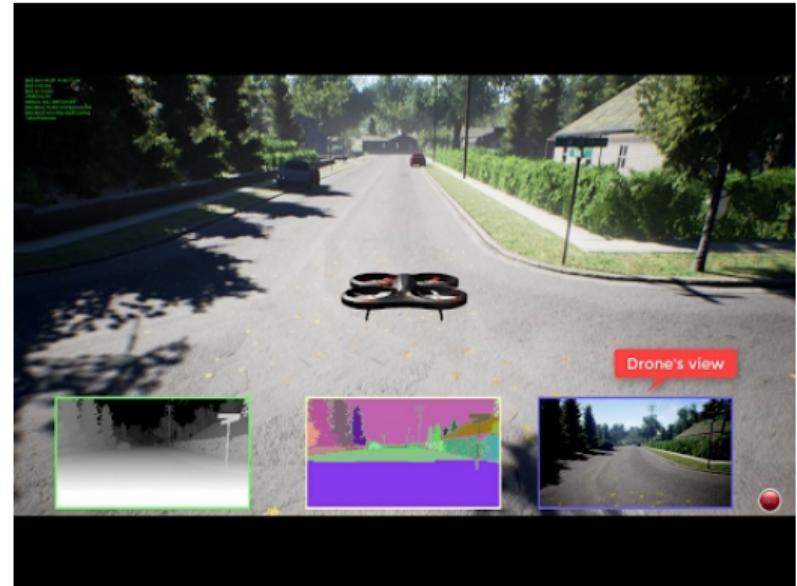


Image source



# Processing



# Terrain Classifier Creation

- Test several methods



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods
  - Combination



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods
  - Combination
- Pre-processing wrappers:
  - Rectification/calibration



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods
  - Combination
- Pre-processing wrappers:
  - Rectification/calibration
  - Downsampling/resizing



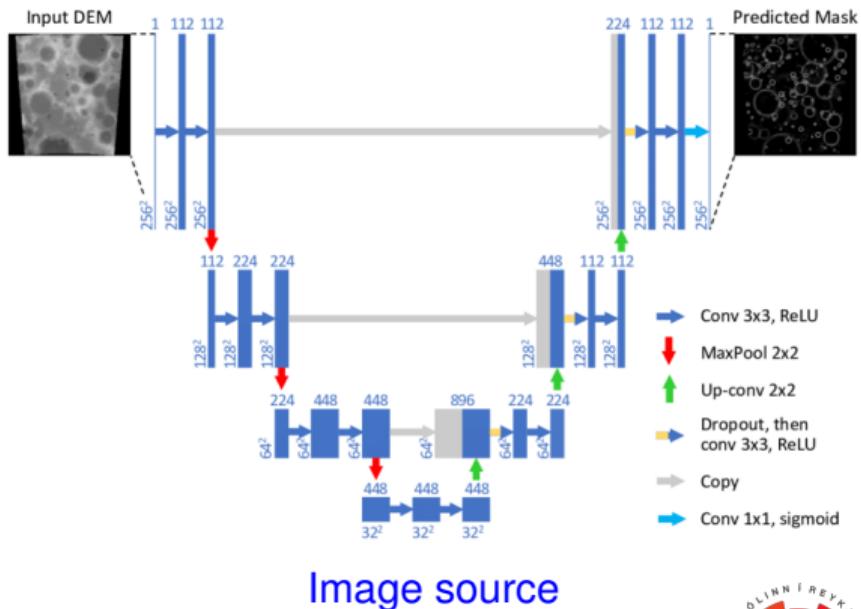
# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods
  - Combination
- Pre-processing wrappers:
  - Rectification/calibration
  - Downsampling/resizing
- Performance comparison
  - Reduce false positives



# Terrain Classifier Creation

- Test several methods
  - Conventional signal/image processing
  - Deep learning methods
  - Combination
- Pre-processing wrappers:
  - Rectification/calibration
  - Downsampling/resizing
- Performance comparison
  - Reduce false positives



# Testing



# Testing in Simulation

- Post-processing wrappers:
  - Safe region tracking
  - Translation to flight commands



# Testing in Simulation

- Post-processing wrappers:
  - Safe region tracking
  - Translation to flight commands
- Integration with ArduPilot/PX4 SITL
- Simulated autonomous missions in AirSim



# Testing in Simulation

- Post-processing wrappers:
  - Safe region tracking
  - Translation to flight commands
- Integration with ArduPilot/PX4 SITL
- Simulated autonomous missions in AirSim
- Qualitative analysis:
  - Does the drone land at all?



# Testing in Simulation

- Post-processing wrappers:
  - Safe region tracking
  - Translation to flight commands
- Integration with ArduPilot/PX4 SITL
- Simulated autonomous missions in AirSim
- Qualitative analysis:
  - Does the drone land at all?
  - Does the safe region tracking work?



# Testing in Simulation

- Post-processing wrappers:
  - Safe region tracking
  - Translation to flight commands
- Integration with ArduPilot/PX4 SITL
- Simulated autonomous missions in AirSim
- Qualitative analysis:
  - Does the drone land at all?
  - Does the safe region tracking work?
  - Does the autopilot software accept the commands?



# Simulation is not enough!



# Testing in the Real World

- Offline
  - Accuracy on real world data

# Testing in the Real World

- Offline
  - Accuracy on real world data
- Lab scenarios
  - Runtime framerate on embedded hardware
  - Power requirements on embedded hardware



# Testing in the Real World

- Offline
  - Accuracy on real world data
- Lab scenarios
  - Runtime framerate on embedded hardware
  - Power requirements on embedded hardware
- Real world landing scenarios



# Drone Upgrades

- Better flight controller: Pixhawk Cube Orange
- Better GPS: Here3

# Drone Upgrades

- Better flight controller: Pixhawk Cube Orange
- Better GPS: Here3
  - Optical Flow
  - LIDAR rangefinder

# Drone Upgrades

- Better flight controller: Pixhawk Cube Orange
- Better GPS: Here3
  - Optical Flow
  - LIDAR rangefinder
- Protective sensor cases, gimbal mounts
  - Intel RealSense D435 RGBD camera
  - Intel RealSense D455 RGBD camera (IMU)
  - Intel RealSense L515 LIDAR (IMU)
  - Texas Instruments IWR6843 60 GHz RADAR

# Drone Upgrades

- Better flight controller: Pixhawk Cube Orange
- Better GPS: Here3
  - Optical Flow
  - LIDAR rangefinder
- Protective sensor cases, gimbal mounts
  - Intel RealSense D435 RGBD camera
  - Intel RealSense D455 RGBD camera (IMU)
  - Intel RealSense L515 LIDAR (IMU)
  - Texas Instruments IWR6843 60 GHz RADAR
- CB: Raspberry Pi 4 + Accelerators



# Main Risks

- The synthetic data does not accurately represent the real world!
  - Show results in simulation.
  - Use real world data → no segmentation masks.



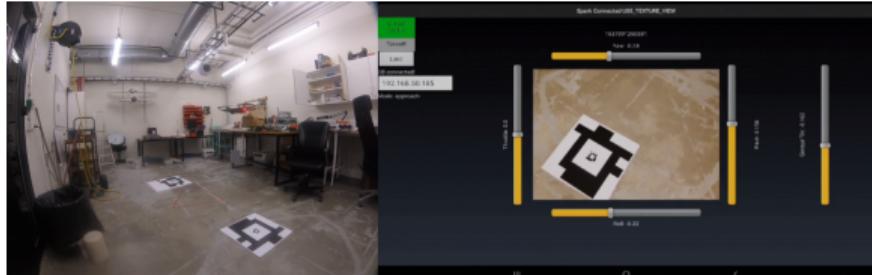
# Main Risks

- The synthetic data does not accurately represent the real world!
  - Show results in simulation.
  - Use real world data → no segmentation masks.
- The embedded hardware is too slow!
  - Reduce computational needs → prune network, decrease input size
  - Use non-embedded hardware → generate reliable flight commands on real world data

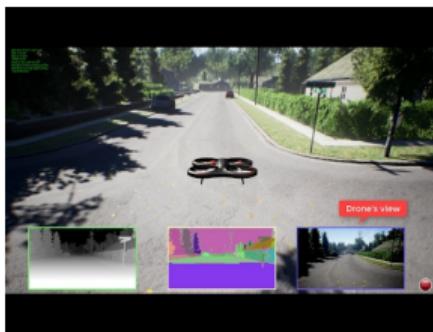


# Summary

- Goal: autonomous drone landing

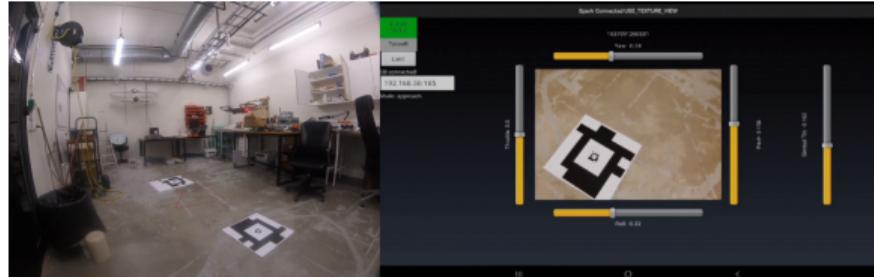


[Click to watch on Vimeo](#)

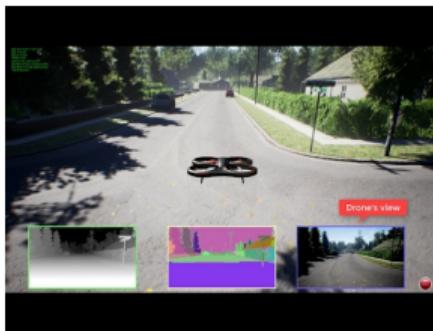


# Summary

- Goal: autonomous drone landing
- Past work: landing via fiducial markers at *known* landing pads
  - Contribution: gimbal-mounted camera setup, new marker variants

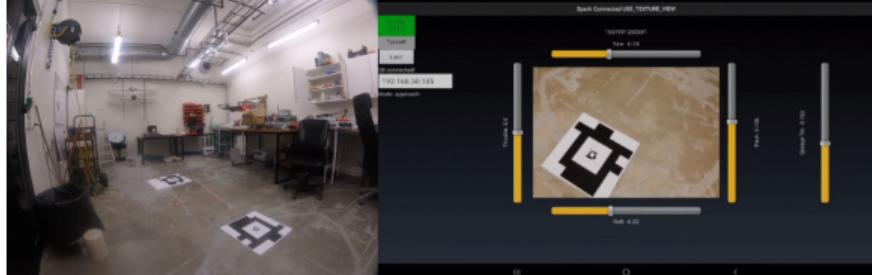


[Click to watch on Vimeo](#)

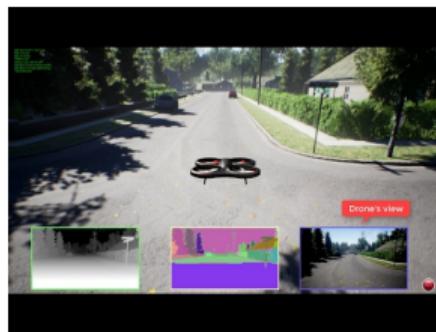


# Summary

- Goal: autonomous drone landing
- Past work: landing via fiducial markers at *known* landing pads
  - Contribution: gimbal-mounted camera setup, new marker variants
- Research plan: unstructured landing
  - Sensors: RGBD, LIDAR/RADAR
  - Topographical/semantic terrain analysis
  - Synthetic data
  - Testing in simulation
  - Real world tests: power/framerate
  - Real world tests: landing with a physical drone

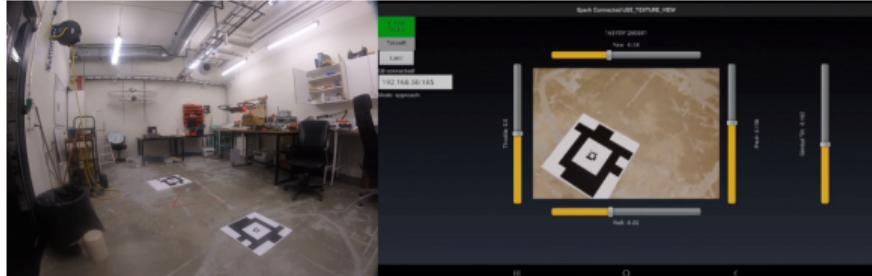


[Click to watch on Vimeo](#)

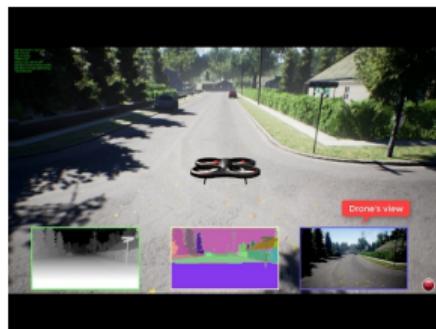


# Summary

- Goal: autonomous drone landing
- Past work: landing via fiducial markers at *known* landing pads
  - Contribution: gimbal-mounted camera setup, new marker variants
- Research plan: unstructured landing
  - Sensors: RGBD, LIDAR/RADAR
  - Topographical/semantic terrain analysis
  - Synthetic data
  - Testing in simulation
  - Real world tests: power/framerate
  - Real world tests: landing with a physical drone
- Thank you for the support!
- Thank you for listening! Are there any questions?



[Click to watch on Vimeo](#)



# Misc

- Google Coral Benchmarks
- Jetson Nano Benchmarks

