AUTONOMOUS DRONE LANDING IN 3D URBAN ENVIRONMENT USING REAL-TIME VISIBILITY ANALYSIS

Oren Gal, Yerach Doytsher, Judah (Udi) Shriki

,

[illegible]

PRESENTAION CONTENTS

INTRODUCTION - CHALLENGES

01

LANDING AUTONOMOUSLY: A CHALLENGE ON FOCUS OF RESEARCHERS

Quadcopters and other types of UAV.
Using Sensors, Shapes or Color, LEDs etc.

02

LANDING SAFELY IS A CHALLENGE FOR EVEN TRAINED PILOTS

Both on manned and remotely controlled aircrafts

03

MACHINE LEARNING TECHNIQUES

Supervised vs. Unsupervised
Creating Data
Comparing Classifiers

INTRODUCTION – GOALS

04

MECHANISM FOR VISUAL PROCESSING

AR Tags Identification and Analysis
Create Large Data

05

SVM TRAINING

Fitting Large Data Into SVM Models and Comparing Classifiers for Accuracy

06

POC SIMULATION

POC Project – Flight Simulation of a Mission

SCOPE OF WORK

- Introduce a Mechanism for Autonomous Landing using Vision
- Compare ML Vs. Calculations
- Simulation Flight POC

SCOPE OF WORK

TARGET POSITION

Known Target Position
Limited Search Area

OBSTACLE AVOIDENS

Path to Target is Clear
No Obstacles

VISIBILITY

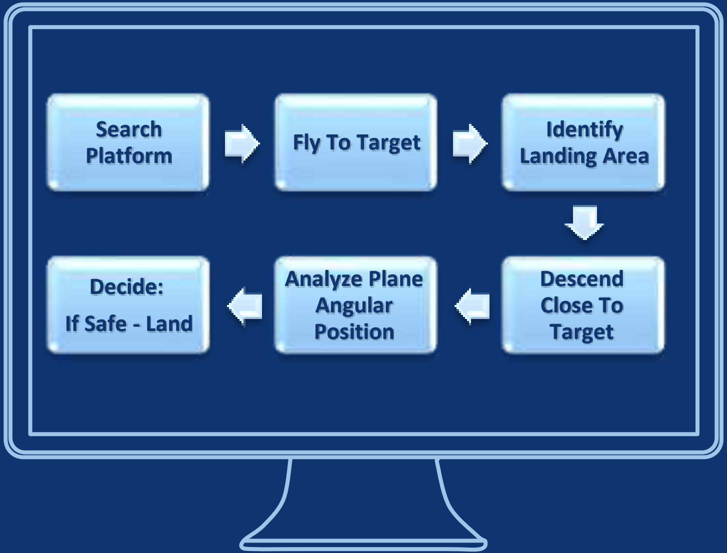
Clear Visibility
Not Obscured

COMMUNICATION

Continuus Comm.
Ground Station Control

PROPOSED MECHANISM

- “Return Home” to navigate close to target
- Then look around until target identified
- Set Course and fly until hovering above target
- Descending and keeping target below
- Final stage – Decision based on Visual Data



ALGORITHMS

Return Home, Search, Navigation

RETURN HOME PROCEDURE

LOST SIGNAL

Automatically starts when signal is lost.
Home set to take-off position

OPERATOR REQUEST

Remote Control Button
Operator Call/Cancel

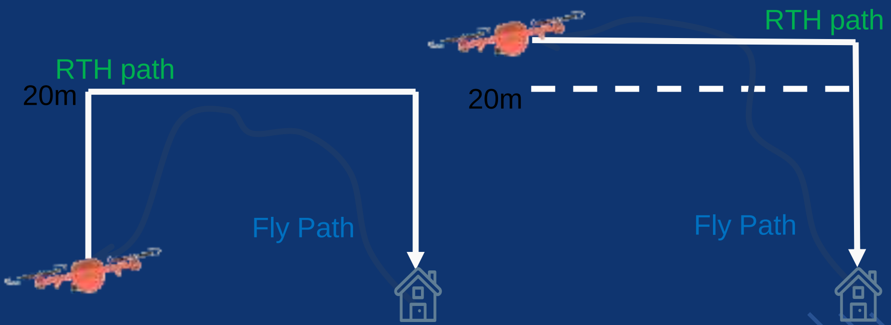
API COMMAND

Programatically
Can also change Home

RETURN HOME PROCEDURE

ALGORITHM

1. If (height < 20m) then go up to 20m
2. Go in a straight line directly to target.
3. On target position, lower height to 2m.

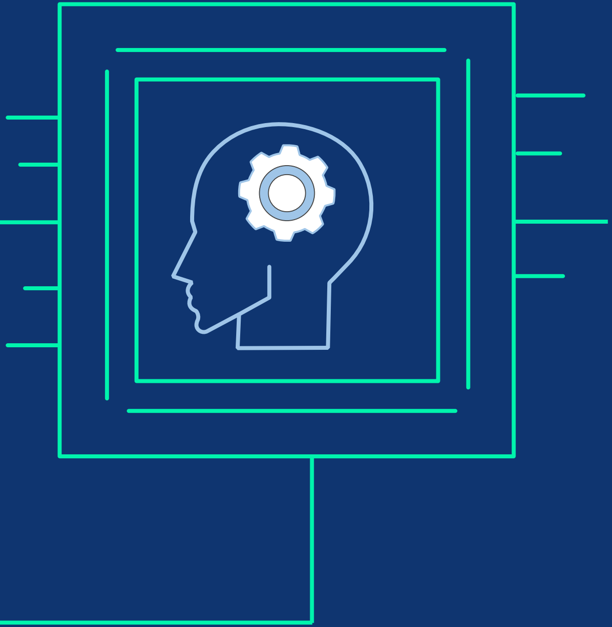


SEARCH STATE

RTH will bring Drone up to a few meters off target.
We assume 1-10m radius of error by GPS accuracy

Goal: find target location visually and set course

SEARCH STATE - ALGORITHM



Search Algorithm:

```
1. height = 4m
2. While (height < max_height):
3.     For (tilt = -20 to -90 step 35):
4.         For (pan = 0 to 360, step 45):
5.             markers = find_markers(image)
6.             If (markers is not None) then: State.next
7.     height.add(1m)
8. State.set(FAIL)
```

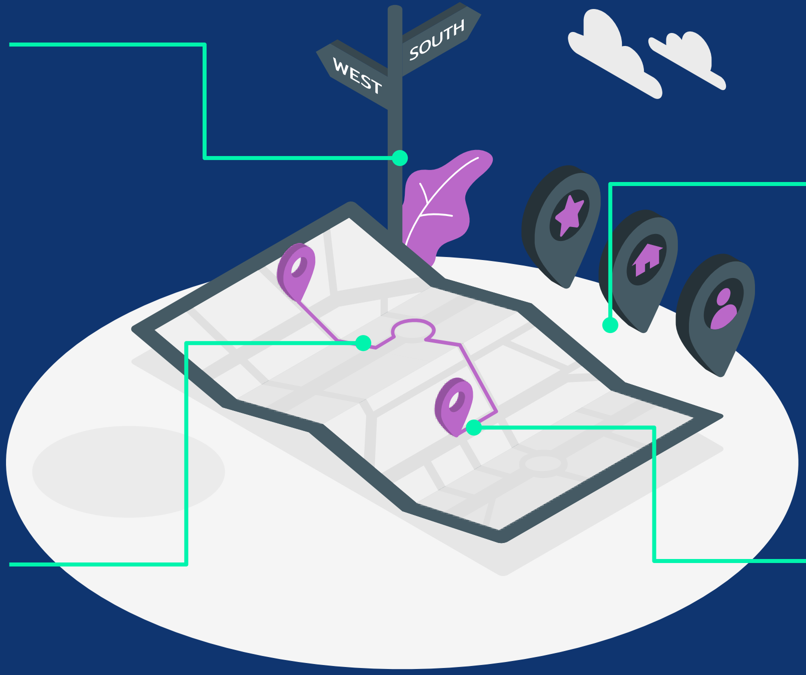
NAVIGATION

MULTIPLE MARKERS

Mean Values of X,Y
Minimum 1 Marker

MARKERS SIZES

Multi resolution
Markers ID series
Distance to speed



NAVIGATE

Keep X close to Zero
Histheresis

FLY FORWARD

Projective View
Forward / Tilt Down

DRONE SELECTION

Model Comparison
Issues in Consideration

DRONE SELECTION – PHASE I

PROS

Can look Downward
Programable + SDK
Only one I had - default

CONS

Old (2012)
Weak Batteries
Low Resolution

A.R. DRONE – STARTING POINT

LIBARDRONE

Using libardrone with OpenCV

- Switching camera not implemented [fixed]
- Require working with obsolete python 2.7
- Worked ok with small load (mission control)
- Not responsive when mission control became complex

PYARDRONE

Replaced with pyardrone

- Python 3 compatible
- Also Did not implement switching camera [fixed]
- Worked OK.

A.R. DRONE – STARTING POINT

GOALS REACHED

Testing Mission Controls:
- State Machine
- Control Drone Flight
- Getting Image Frames

Testing Image Processing:
- Identify Markers
- Trigger Operations

Testing ML:
- Create Initial Dataset
- Test Classifier

BEBOP 2

- Fish-eye Lens Camera
- Digital 3-axis stabilization
- Digital pan/tilt 180°
- Strong 6” propellers
- 2km Range (using Skycontroller)
- Up to 30 min. flight time
- Supported in Sphinx simulator
- No more hardware debugging!

PROGRAMING BEBOP2

AR.SDK 3

Next Generation SDK
for Parrot Drones

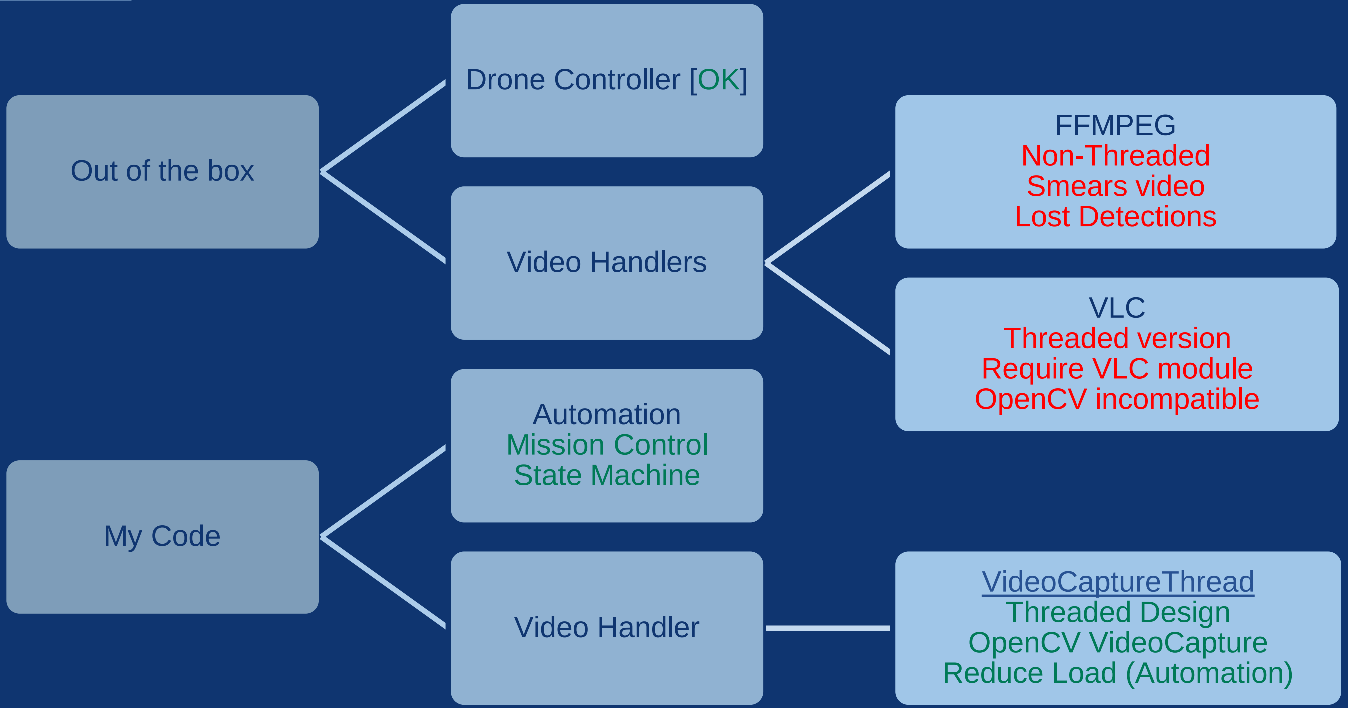
OLYMPE

A Parrot Python Package
Part of Ground SDK
Closed Virtual Environmet
Cannot Be Adapted or Changed

PYPARROT

Third Party Python Package
Encapsulate AR.SDK3
Edit and Add Features
Run/Debug from any Python IDE
Integrates with other packages
Supported Threading Video

PYPARROT ADAPTATION



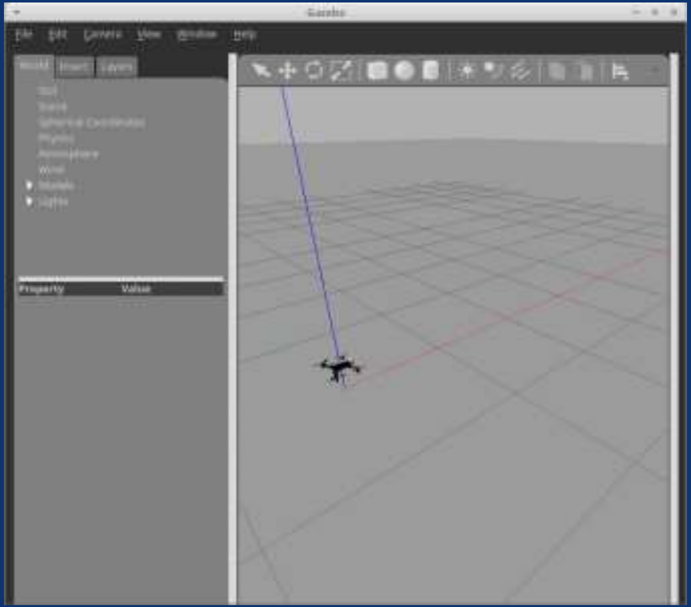
SPHINX SIMULATOR

GAZEBO

Based on Gazebo framework

CONTROLS

Operable with Controllers, Application



FIRMWARE

Official Parrot Firmware

MODELS

Official Parrot Drones Graphics Models Physics Models

MACHINE TRAINING

Creating Data
Fitting Large Data-set
Comparing Classifiers
Improving Accuracy

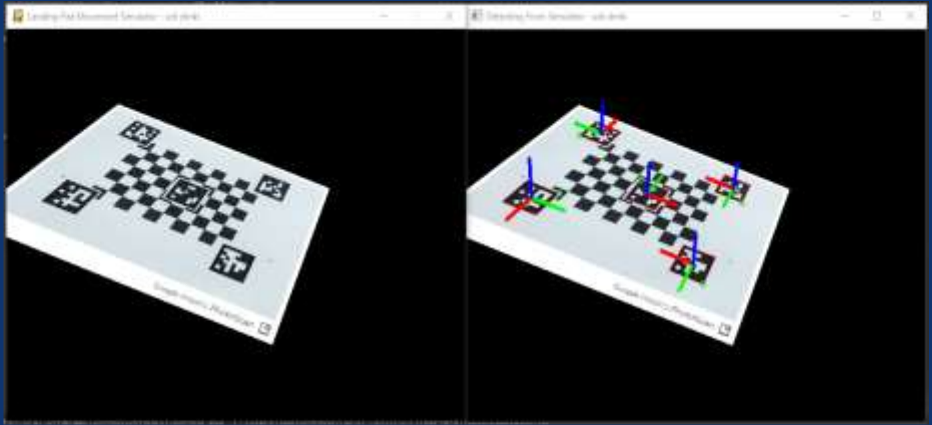
DATA CREATION

AUTOMATION WITH OPENGL

OpenGL Simulation output used as camera input to existing code
Controllable and precise to accurately label data vectors
Could run on separate threads and even different machines (parallel)

RESULTS

A few days to create (automatic)
Dataset of 15M vectors
accurate labeling (?)



FITTING AND COMPARING CLASSIFIERS

- Dataset of 15M vectors for training
- Fitting all-at-once could not be performed
  Solution: Partial fitting (details) ℹ
- Testing against different classifiers, parameters
- Improving bad results:
  - Classifier of Classifiers results (Smart Voting) – no Improvement ☹
  - Rectify errors of visual detections – Imroved ☺

Best Results – SGD Classifier with loss='log'

COMPARING CLASSIFIERS

- First results [FAIL]

- Classifier of Classifiers [FAIL]
  - 10 best classifiers
  - Vector of results
  - Voting (not fair)
  - Best result 76.8%

- Errors in Detections?

Classifier Type	Accuracy
SGD, epsilon insensitive	57.341%
SGD, hinge	75.716%
SGD, huber	59.841%
SGD, log	73.658%
SGD, modified huber	73.362%
SGD, squared eps. insensitive	59.6%
SGD, squared hinge	73.857%
SGD, squared loss	57.171%
Perceptron	74.579%
Bernoulli NB	62.317%
Passive Aggressive Classifier	74.455%

COMPARING CLASSIFIERS

- Used Calculations over data to rectify extreme errors

Drastic Improvement in accuracy!

Classifier Type	Best	Original	diff
SGD, epsilon insensitive	83.450%	57.341%	26.11%
SGD, hinge	85.062%	75.716%	9.35%
SGD, huber	81.876%	59.841%	22.04%
SGD, log	86.175%	73.658%	12.52%
SGD, modified huber	86.131%	73.362%	12.77%
SGD, sqr-eps. insensitive	82.019%	59.6%	22.42%
SGD, squared hinge	85.891%	73.857%	12.03%
SGD, squared loss	82.942%	57.171%	25.77%
Perceptron	85.470%	74.579%	10.89%
Bernoulli NB	81.664%	62.317%	19.35%
PassiveAggressive Classifier	84.041%	74.455%	9.59%

SUMMARY

First Review Amendments ℹ
Experiments
Future Work

EXPERIMENTS AND RESULTS

UNIT TESTS

- Search – find visual marker [OK]
- Found – set course [OK]
- Fly – move and keep course [OK]
- Descend – lower height while keep target below [OK]
- Decide – use visual data to decide when it is safe to land:

    1. Classifier tested separately [OK]
    2. Some problems during simulation – could be solved. (sizes/ distortions?)

EXPERIMENTS

- Fully tested only with manually rotating landing pad in simulation
- Improved landing pad detection from afar using multiple marker sizes

FUTURE WORK

WAVES

Experiments with full simulation of waves

DRONE

Experiment with real live drone flight

PATH

Add Path planing and obstacle avoidence

FIRMWARE

Integrate with flight computer for full autonomous UAV

CLASSIFIER

Fix classifier integration into the mechanism

THANKS!