# Vision-based Autonomous Landing in Catastrophe-Struck Environments

*Authors:* Mayank Mittal, Abhinav Valada, Wofram Burgard

---

Michele Cipriano

January 27, 2019

Control Problems in Robotics: Modeling and control of multi-rotor UAVs
Department of Computer, Control and Management Engineering
Sapienza University of Rome

# Introduction

Equipping UAVs with **bioradars**:

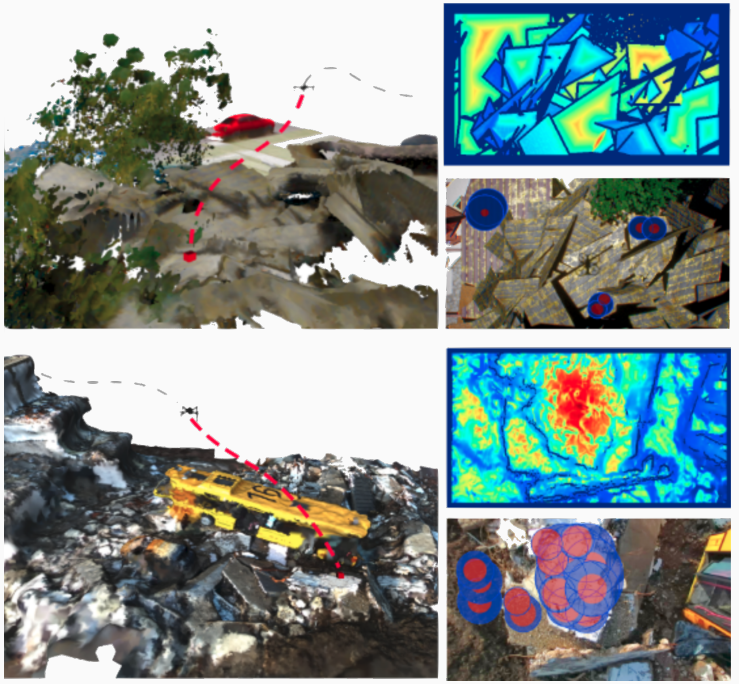
- estimate a set of candidate **safe landing sites**
- optimize over the global area via **clustering**
- project landing sites onto **volumetric reconstruction**
- compute a **minimum-jerk trajectory** to land on debris piles
- validate on simulated and real-world environments

# Technical Approach: State Estimation

- **ORB-SLAM2** (oriented FAST and rotated BRIEF)
- downward-facing stereo camera
- multi-sensor fusion (EKF) using IMU, barometer and GPS
- sensor precalibrated with Kalibr toolbox

# Technical Approach: Landing Site Detection

- Depth Information $J_{DE}$
- Flatness Information $J_{FL}$
- Steepness Information $J_N$
- Energy Consumption Information $J_{EC}$
- k-d tree + agglomerative hierarchical clustering algorithm



# Technical Approach: Landing Site Detection

Confidence in **Depth Information** $J_{DE}$:

$$J_{DE}(p) = 1 - \frac{D(p)^2 - min\{D^2\}}{max\{D^2\}}$$

with $D$ depthmap obtained from the stereo camera and $p = (x, y)$ pixel in the depthmap.

# Technical Approach: Landing Site Detection

**Flatness Information $J_{FL}$:**

$$di(B, p) = min\left\{ \|p - q\| \middle| B(q) = 1 \right\}$$

$$J_{FL}(p) = di(Canny(D), p)$$

with $B$ binary image and $p, q$ pixels in the image plane. *Canny* applies the **Canny edge detector** over the depthmap $D$.

# Technical Approach: Landing Site Detection

### Steepness Information $J_N$:

- point cloud from the depthmap in global frame
- **average 3D gradients algorithm** to estimate normals map $N$
- evaluate deviation of the normalized surface normal $\hat{n}$ wrt z-axis $\hat{z}$ in the world frame: $\theta = cos^{-1}(\hat{n}^T\hat{z})$
- compute steepness score for each pixel $p$ given $\theta_{th} = \pi/12$ maximum tolerable slope

$$J_N(p) = exp\left\{-\frac{\theta^2}{2\theta_{th}^2}\right\}$$

# Technical Approach: Landing Site Detection

**Energy Consumption Information** $J_{EC}$:

$$J_{EC}(p) = \int_{t_0}^{t_f} P(t)dt$$

with $t_0$ and $t_f$ time of flight to reach $p$ and $P(t)$ instantaneous battery power. Approximate the integral with **Euclidean distance** between the UAV and $p$.

# Technical Approach: Landing Site Detection

Scale costmaps to the same range through **min-max normalization** and compute the final decision map $J$ taking a weighted sum:

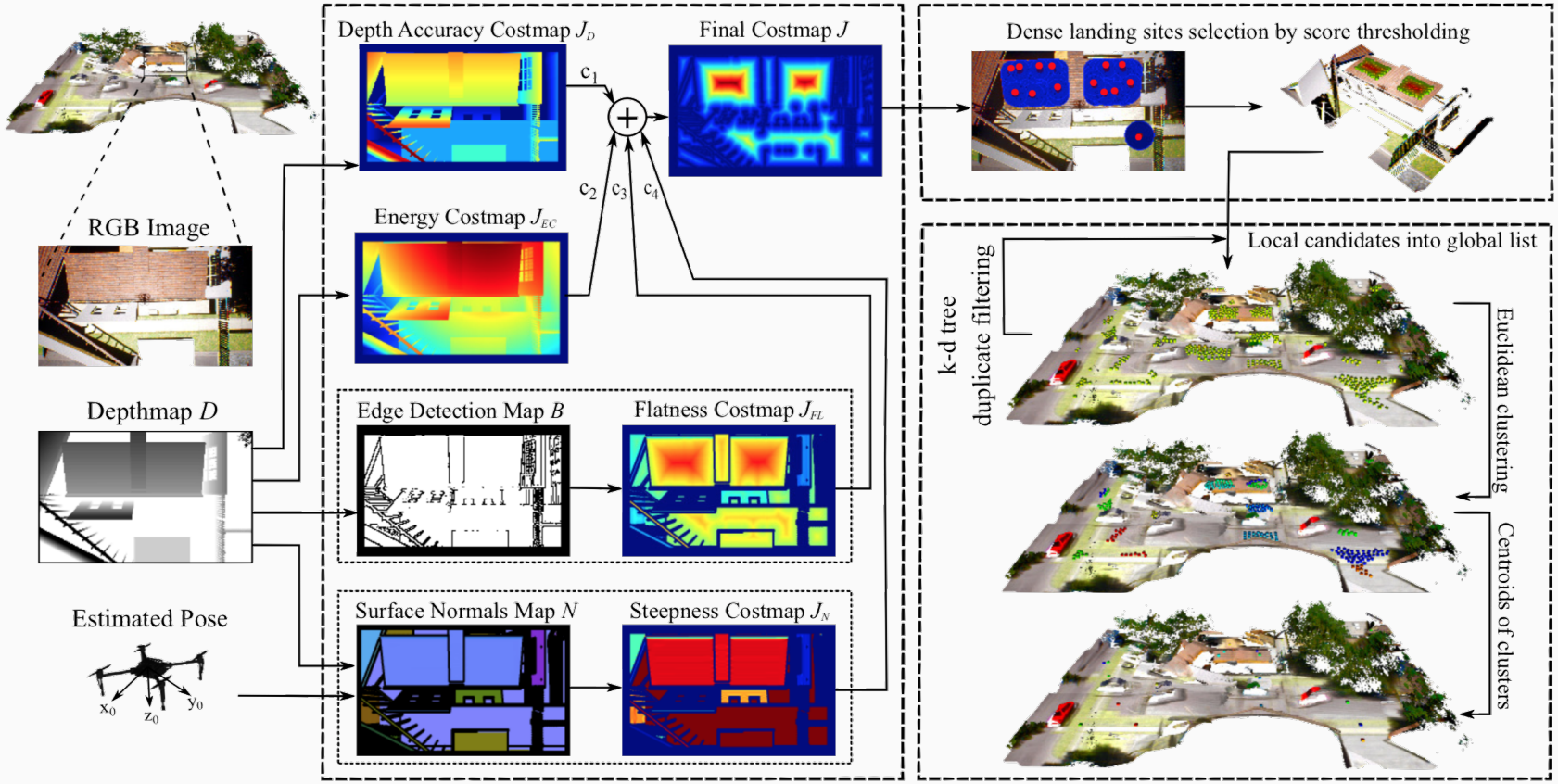
$$J = c_1 J_{DE} + c_2 J_{FL} + c_3 J_N + c_4 J_{EC}$$

$$c_i \in [0, 1] \quad \sum_i c_i = 1$$

- keep the sites checking whether the UAV could actually land
- **k-d tree** to efficiently store new landing sites
- **hierarchical clustering** algorithm to agglomerate sites

# Technical Approach: Landing Site Detection

**Figure 1:** Overview of the landing site detection algorithm. In the costmaps, red indicates high score while blue indicates a lower score. Detected landing sites are projected onto a 3D reconstruction of the environment.



# Technical Approach: 3D Volumetric Mapping

Probabilistic volumetric map:

- navigation and planning
- low-resolution map (0.5m) using **OctoMaps**
- faster trajectory planning

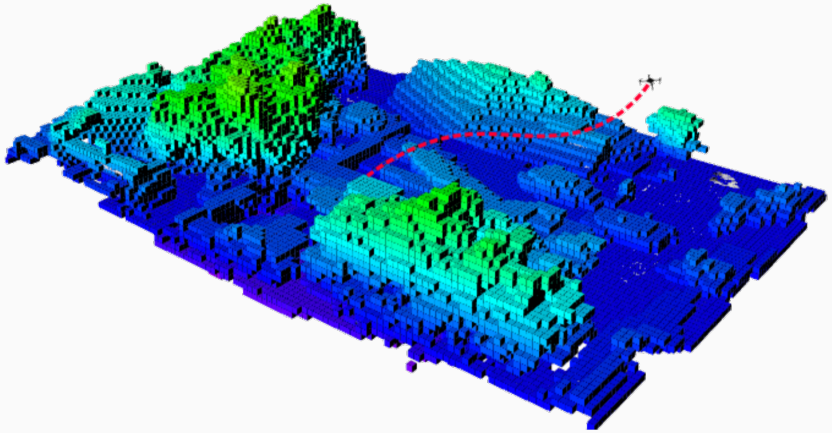
3D textured mesh:

- dynamic map growing using **Voxblox**
- high-resolution mesh transmitted to the rescue team

# Technical Approach: Landing Trajectory Estimation

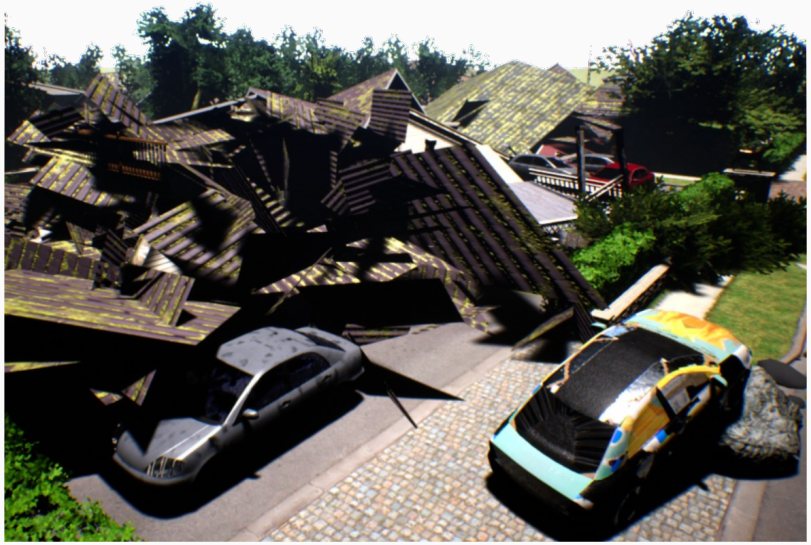
Minimum-jerk trajectory generator with non-linear optimization:

- **RRT*** for collision free path
- line-of-sight for waypoints
- minimum-snap polynomial trajectories using differential flat model
- high speed arcs in obstacles free regions
- low velocities in tight places



# Experimental Evaluation

## Hyperrealistic Simulation



- AirSim + Unreal Engine + ROS
- RGBD 640x480 20Hz
- Octomap 0.5m
- Voxblox 0.1m

$c_1 = 0.05, c_2 = 0.4, c_3 = 0.4, c_4 = 0.15$

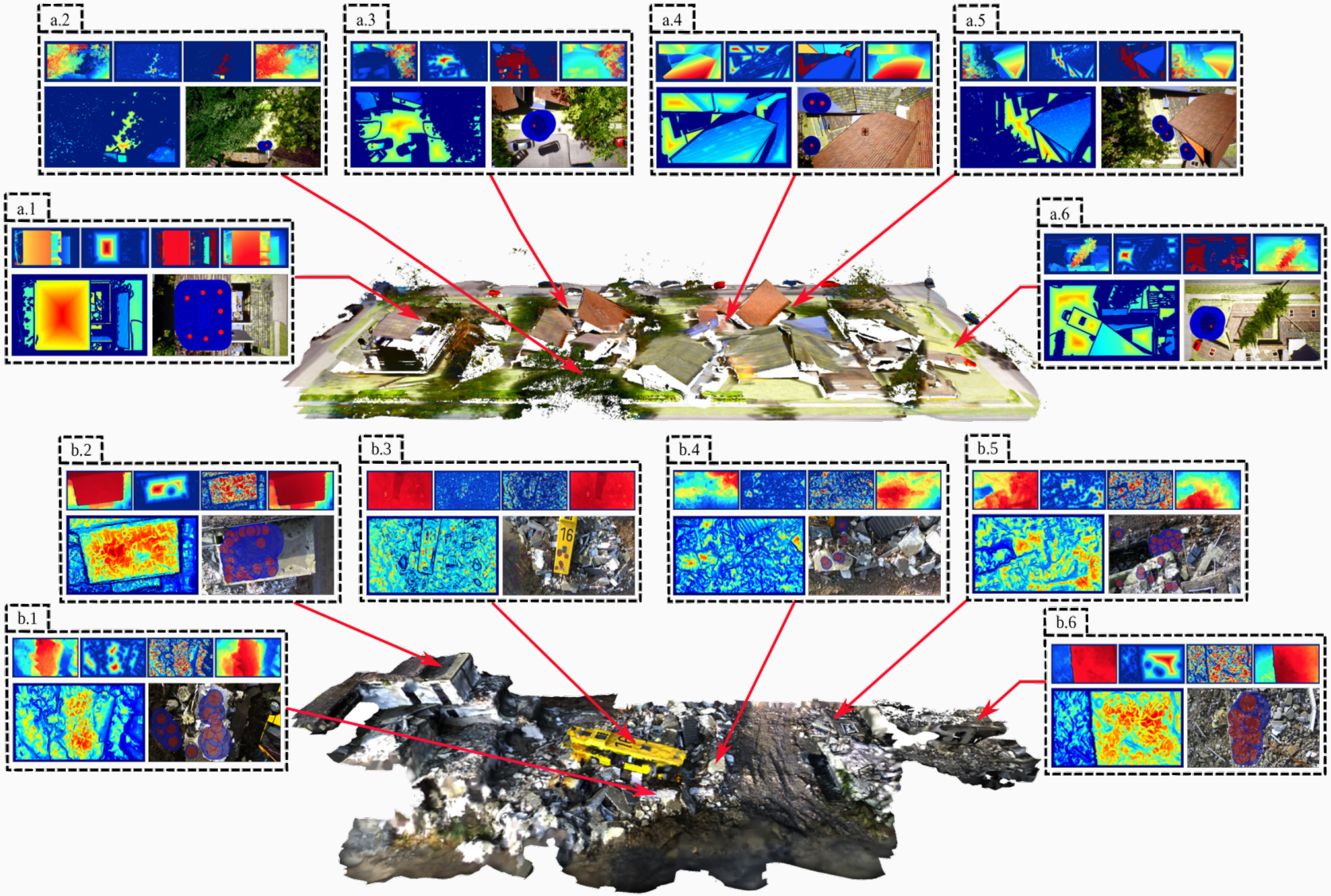
## Training Center for Rescue

- DJI M100 + NVIDIA TX2
- ZED stereo camera 640x480 20Hz
- Octomap 0.5m
- Voxblox 0.1m

$c_1 = 0.15, c_2 = 0.35, c_3 = 0.4, c_4 = 0.1$

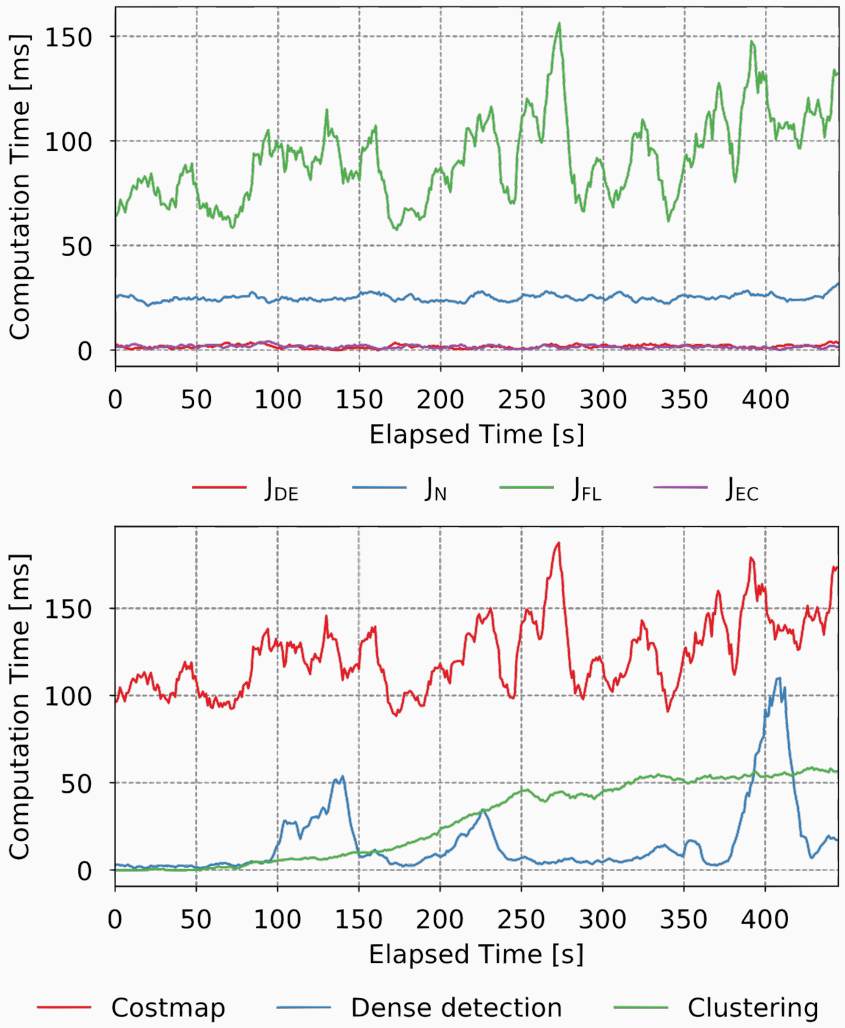
# Experimental Evaluation

**Figure 2:** Costmap evaluation and dense landing sites detection steps in both simulated and real-world scenarios.



# Computation Costs

- landing site detection algorithm runs in **167.5ms** using **193.8MB**
- $J_{DE}$, $J_N$, $J_{EC}$ are linear
- $J_{FL}$ depends on distance transformation operation: $O(dk)$, $d = 2$
- clustering time and memory complexity: $O(n^2)$ and $O(n)$



# Conclusion

- **vision-based autonomous landing system** for UAVs
- bioradars, search and rescue operations
- hazardous terrain factors
- nearest neighbor filtering and clustering
- both low and high resolution map

Q&A

# References


M. Mittal, A. Valada, and W. Burgard, “Vision-based autonomous landing in catastrophe-struck environments,” *CoRR*, vol. abs/1809.05700, 2018.