Robust FPGA-centric CNN visual localisation for GPS-denied UAVs

Max Gadd
Supervised by Matthew D’Souza

Problem and Motivation

Global Positioning System (GPS) signals can be obstructed, interfered with, or deliberately denied, leaving small unmanned aerial vehicles (UAVs) without a reliable global reference. Visual localisation provides a camera-based alternative, but performance degrades under illumination, seasonal, and viewpoint changes. Large deep models that improve robustness are often impractical for small aircraft because of limits on size, weight, and power.

This project targets real time, on board, absolute localisation from a single image on low power embedded hardware. A convolutional neural network (CNN) maps a monocular frame directly to coordinates in a pre surveyed reference image, avoiding continuous feature tracking and cloud offload. The method is intended to complement inertial devices such as an inertial measurement unit (IMU) and a barometer by providing periodic absolute fixes that bound drift in an inertial navigation system. The goal is to enable reliable autonomy when satellite navigation cannot be trusted, while keeping latency and energy use within the strict budgets of small UAV platforms.

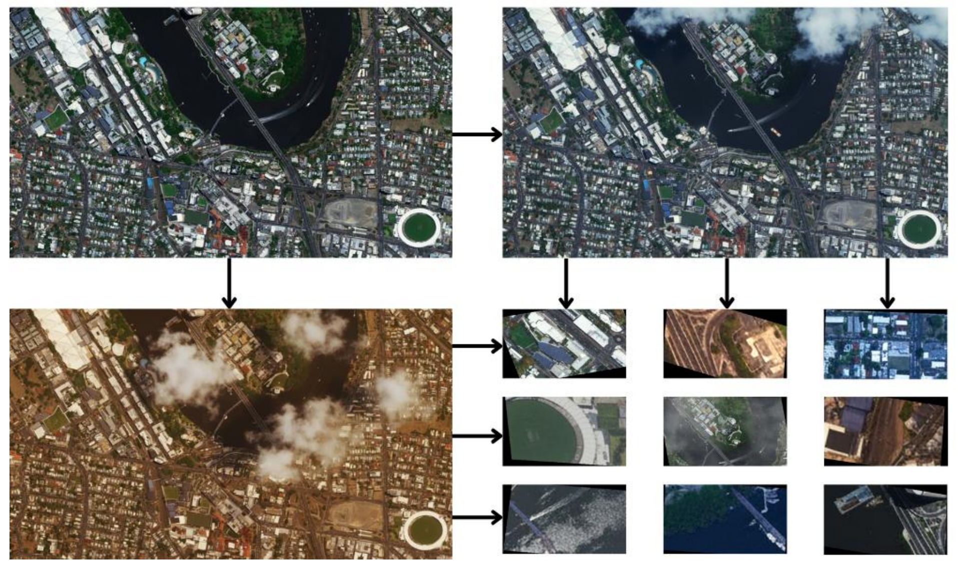


Figure 1: **Example dataset and training samples generation using a single satellite image.** A source image (*Top Left*) can be augmented using the Google Flash 2.5 image generation model and custom prompting to generate images such as the *Top Right* and *Bottom Left* images. These augmented images add diversity to the dataset, to help create varied training samples (bottom right).

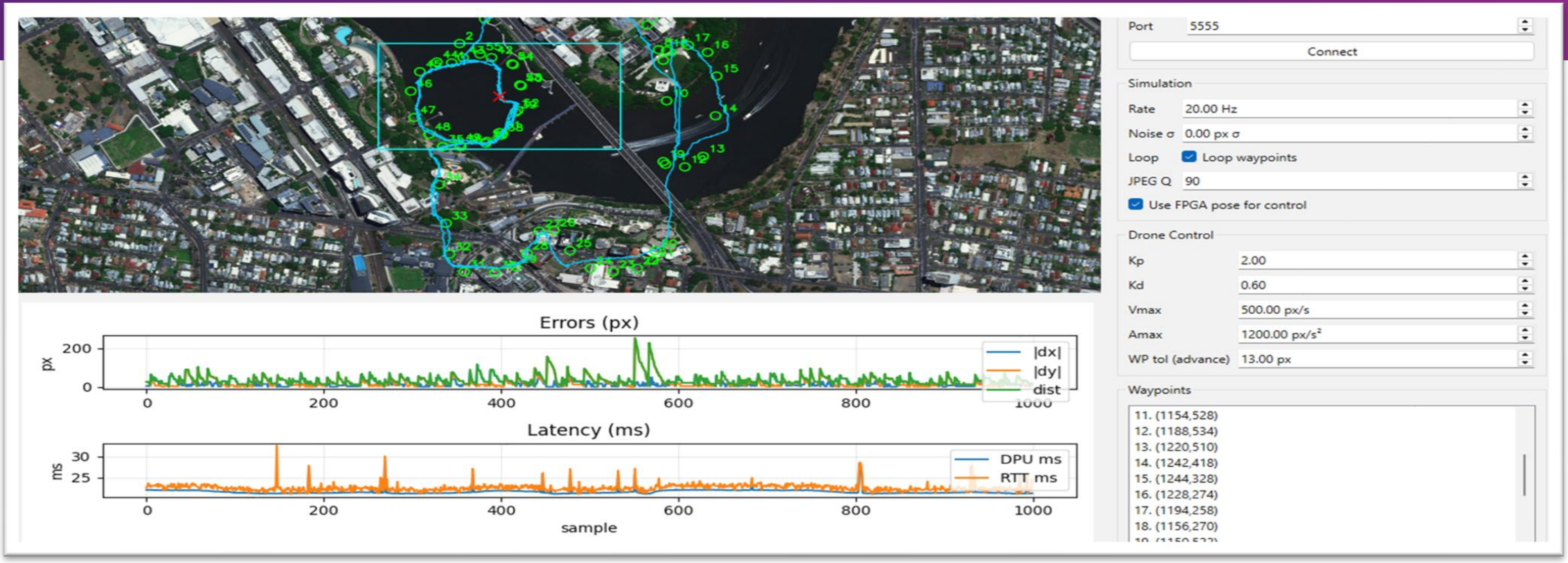
Approach and System Design

A compact convolutional neural network (CNN) was designed to regress normalised image coordinates (x, y) within a pre surveyed map from a single monocular frame. Training used PyTorch with extensive augmentation, including exposure, blur, and weather transforms, AI generated variants seeded from original images, and altitude, also called zoom, sweeps to vary field of view. The trained model was exported to the Open Neural Network Exchange (ONNX) format, quantised to 8-bit integer (INT8), and compiled with AMD Vitis AI for the AMD Kria KV260 Deep Learning Processing Unit (DPU B4096).

At runtime, the ARM processing system performed pre and post processing while the DPU accelerated convolutions to achieve low latency inference fully on board. The network outputs normalised coordinates that are scaled to pixel locations in the reference image. These absolute estimates are designed for fusion with an inertial measurement unit and a barometer, combining high-rate inertial data with periodic absolute visual fixes. A Python based interface connects to the field programmable gate array (FPGA), streams images, and logs predictions to support repeatable experiments and live demonstrations.

Max Gadd mgadd02@gmail.com

School of Electrical Engineering and Computer Science



Port 5555
Connect

Simulation
Rate 20.00 Hz
Noise σ 0.00 px σ
Loop [✓] Loop waypoints
JPEG Q 90
[✓] Use FPGA pose for control

Drone Control
Kp 2.00
Kd 0.60
Vmax 500.00 px/s
Amax 1200.00 px/s^2
WP tol (advance) 13.00 px

Waypoints
11. (1154,528)
12. (1188,534)
13. (1220,510)
14. (1242,418)
15. (1244,328)
16. (1228,274)
17. (1194,258)
18. (1156,270)

Figure 2: FPGA Flight Emulation Tool. This figure depicts a sample usage of the flight emulation tool which communicates to the KV260 board over an ethernet connection. This python interface allows the placement of custom waypoints on a test image, and the ability for a user to specify PID controls for the drone simulation. The path travelled compared to the optimal path can be taken and examined to reveal information regarding the accuracy of such a navigation system, as well as the precision. It can be observed that multiple iterations of the same waypoint mission can lead to different paths travelled, most likely due to the tuneable parameters, as the model is deterministic in nature. The latency of the DPU to process the given image is plotted alongside the ethernet RTT latency, Simulated error is also plotted.

Altitude-aware training

Higher altitude, zoomed out imagery reduces mean pixel error by providing more scene context for absolute localisation.

Altitude aware training improves accuracy as altitude increases. Higher altitude exposes a wider field of view, giving the CNN more global context to anchor absolute position. During evaluation, mean pixel error decreases consistently with increasing altitude or zoom out. The training set includes explicit altitude sweeps and AI generated variants to teach scale and appearance invariance. This effect is summarised in Figure 4 with mean, median and 90 percent confidence intervals across validation scenes.

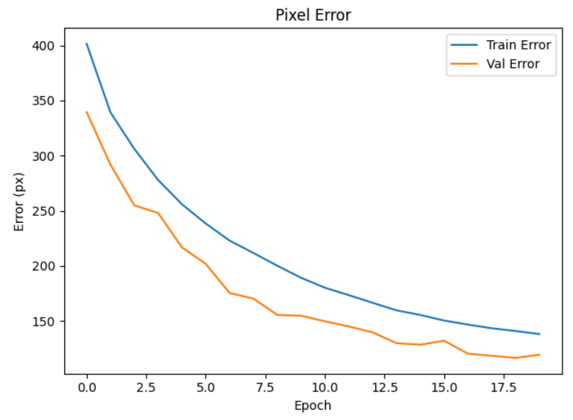


Figure 3: Pixel error during training on Full HD base image. For a given GPS located base image of Full HD resolution (1920x1080 pixels) during training, the validation set recorded the lowest error achieved to be approximately 125 pixels by cartesian distance. This is given the random training inputs of resolution (426x240) which are what would be provided by a drone’s camera. These resolutions could all be upscaled for a likely accuracy gain, and a large sacrifice to hardware throughput. The error of an estimation is taken between the real position and estimated centre location of the camera image. From this we can conclude for any given input image, we can reasonably expect an error measurable by approximately 5% of the diagonal dimension.

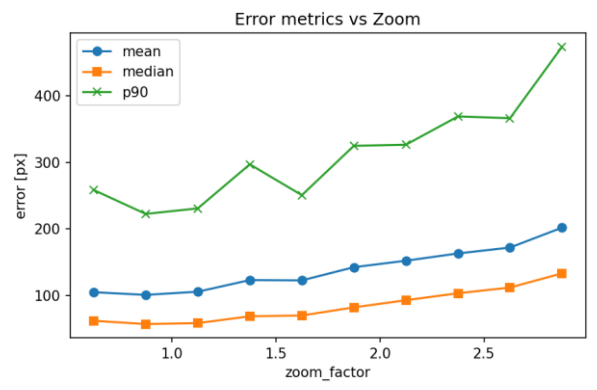


Figure 4: Pixel error by zoom factor (altitude). To simulate the effects of images taken at a variety of altitudes, a zoom factor is applied to all generated dataset sub-samples. These sub-samples are what are depicted in the *Bottom-Right* of Figure 1. The zoom factor clearly augments the images in a way that emulates the varied altitude of a drone or other aerial vehicle. Ultimately, correlating the zoom factor alongside prediction error reveals that lower altitude images yield a worse prediction This is intuitive as higher altitude images are likely to contain more significant image features and landmarks.

Acknowledgements

Supervision by Dr Matthew D’Souza is gratefully acknowledged. Technical staff from the School of Electrical Engineering and Computer Science provided essential hardware and laboratory support.

On-board demo interface

UAV motion is emulated, and plots predicted flight paths on a reference map for repeatable demonstrations.

A lightweight Python interface connects to the FPGA, emulates UAV motion, and visualises the predicted flight path. The tool streams camera frames, pushes them through the on board DPU, and plots the resulting coordinates onto a reference map. This enables controlled navigation what if tests without flying a vehicle, and it provides a clear, repeatable demonstration for examiners. The same interface captures logs for offline analysis and poster figures.

Results, Conclusions, and Next Steps

On device inference sustains roughly 32 frames per second with total board power below 11 W on the Kria KV260 platform. Mean pixel error decreases as altitude increases, confirming that wider context improves absolute localisation. Quantisation aware training closes much of the gap to a full precision baseline while preserving throughput. Against an Oriented FAST and Rotated BRIEF with Random Sample Consensus prototype, the model is more robust to illumination and seasonal change on the project dataset without requiring frame to frame tracking or external compute. The system is designed to complement inertial devices: periodic absolute fixes bound inertial drift and can be fused with an inertial measurement unit and a barometer. Planned work includes mixed precision to protect the most sensitive operations, lightweight temporal cues to stabilise difficult frames, on airfield validation, and evaluation of map update and loop closure strategies for longer deployments. The results indicate that an INT8 CNN on a DPU can deliver practical, low power localisation for small UAVs when GPS is unreliable.