UASys

PROJECT ARTEMIS

Efficient Visual Avoidance and Control for UAVs on Embedded GPUs

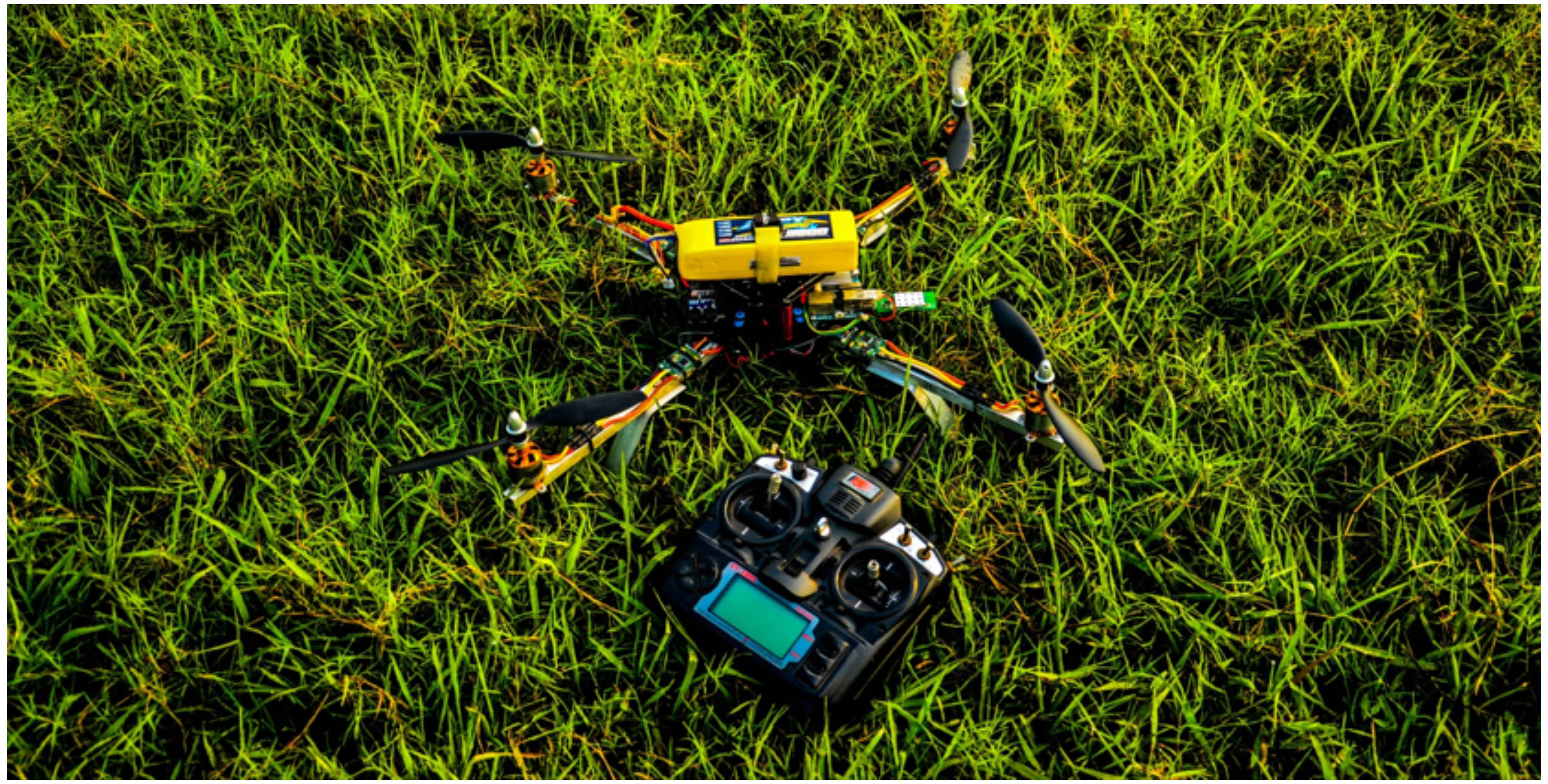
Mohammed Kabir

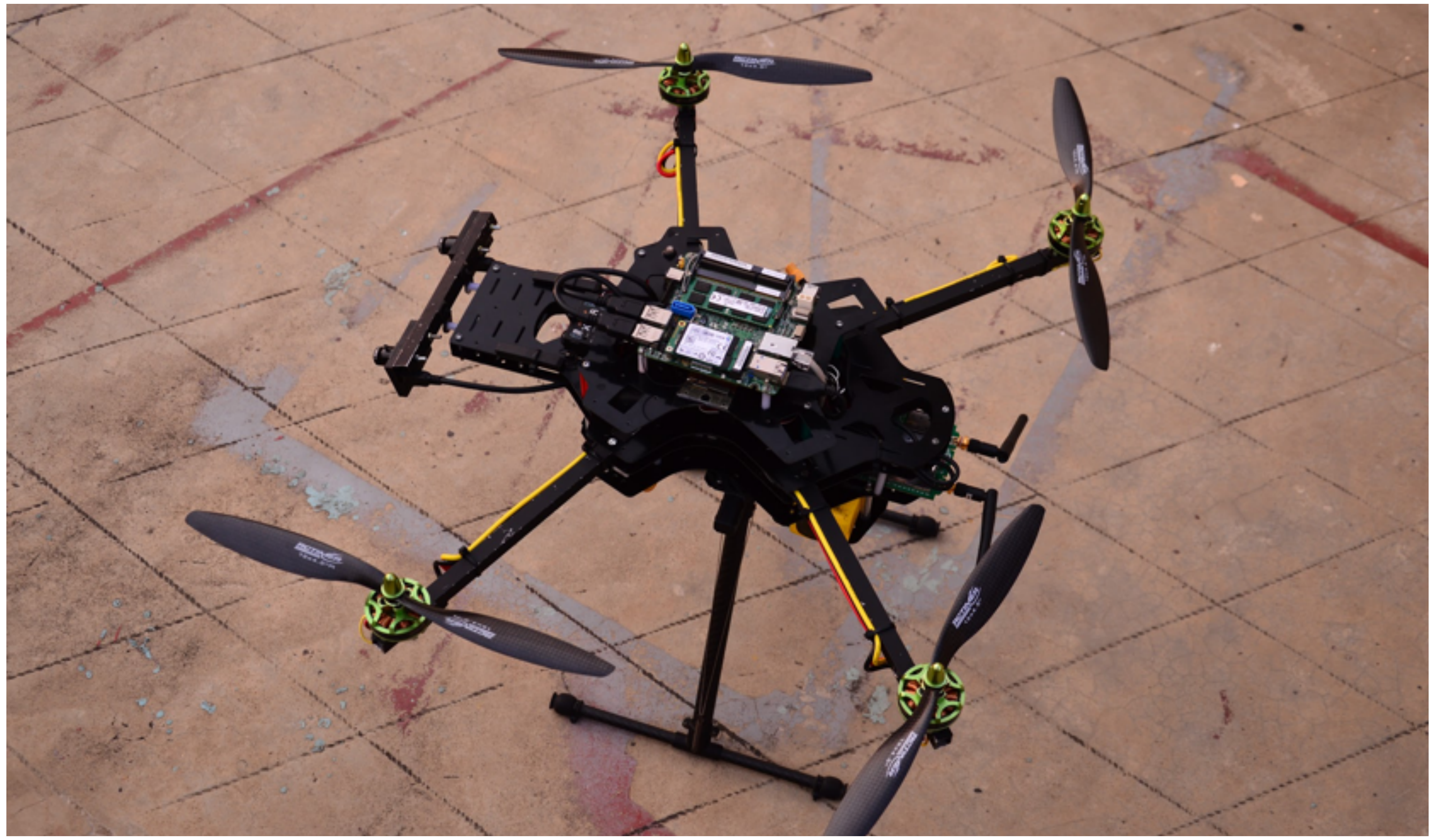
PROJECT ARTEMIS – A HISTORY

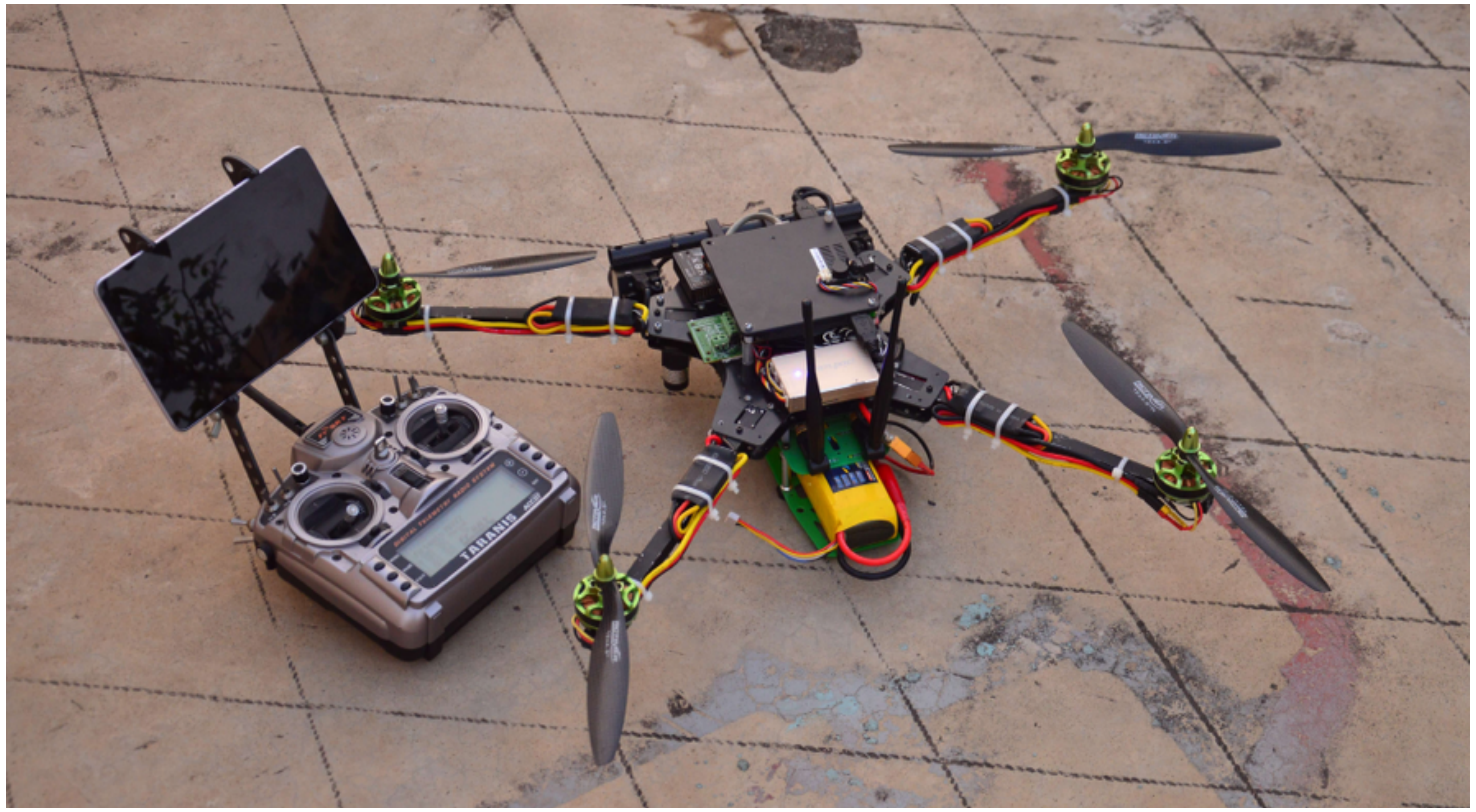
- Early 2014 - Started as a small research project of mine to allow safe indoor drone control.
- Mid 2014 - First successful flights with optical flow based positioning.
- Late 2014 - Started research into ‘active’ vision based state estimation.
- Early 2015 - First flights with monocular visual odometry for state estimation.
- Mid 2015 - Requisite funding collected, design of MAV2 with multi-camera visual odometry and stereo-vision started.
- Late 2015 - MAV2 completed, highly successful flights with loosely coupled vision IMU fusion.
- Early 2016 - MAV3 completed, with tightly coupled stereo-inertial odometry, stereo based depth mapping, autonomous obstacle avoidance.
- Now - Artemis MAV5 flying with GPU computing, robust stereo-inertial odometry with GPS fusion and obstacle avoidance. Field tests with industry partners in progress.

PREVIOUS WORK

- Clockwise from the top are our MAVs in chronological order.
- The development cycle optimises for size and reduces power requirements for every new generation.





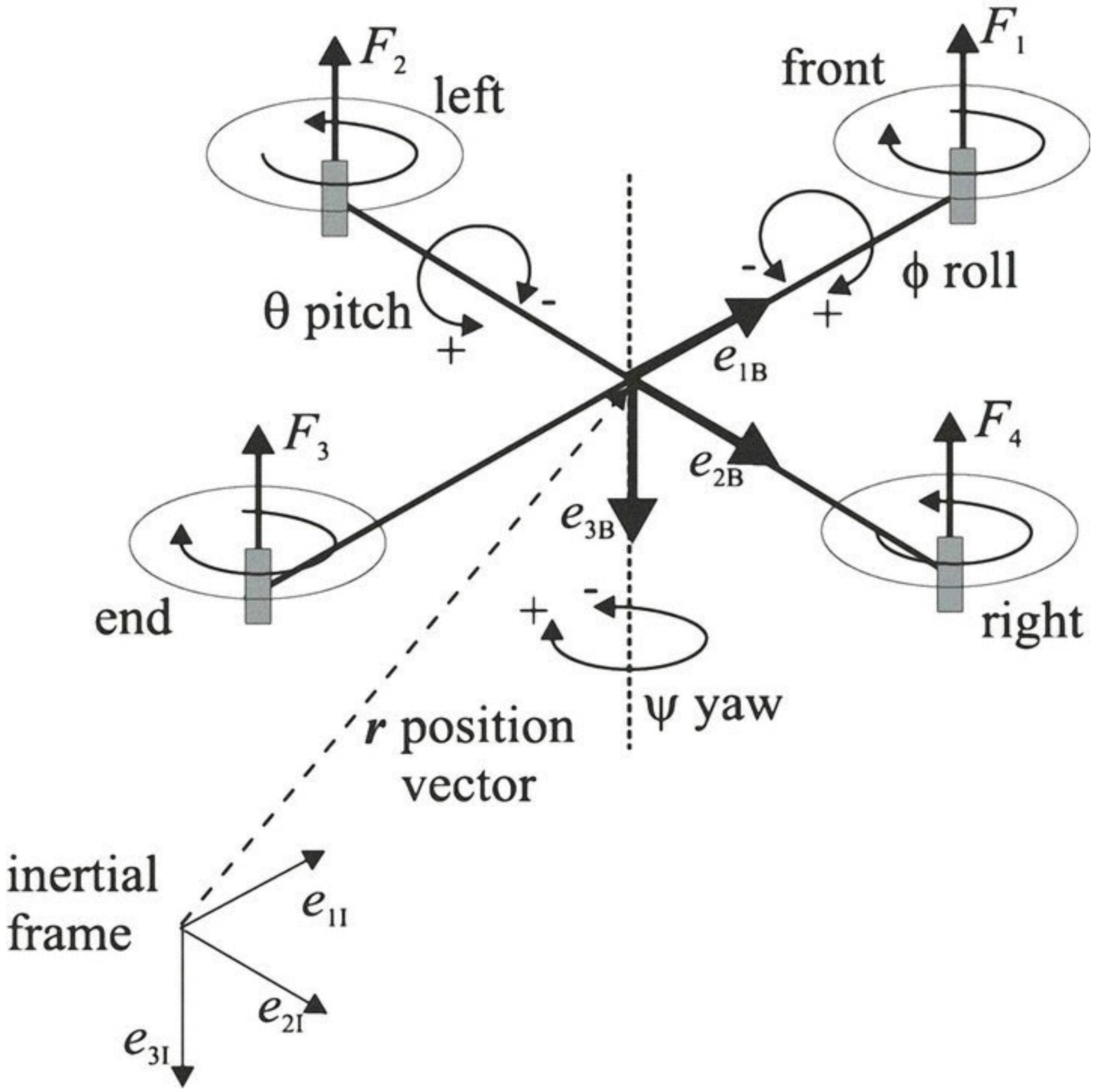


STATE OF THE INDUSTRY

- Highly dependent on GPS in assisted modes.
- Requires sufficient piloting skills in non GPS-assisted modes.
- Chance of ‘flyaways’ due to bad GPS reception.
- Immediate need for robust GPS-agnostic navigation methods.
- Obstacle avoidance is only marginally effective.
- Not truly smart.

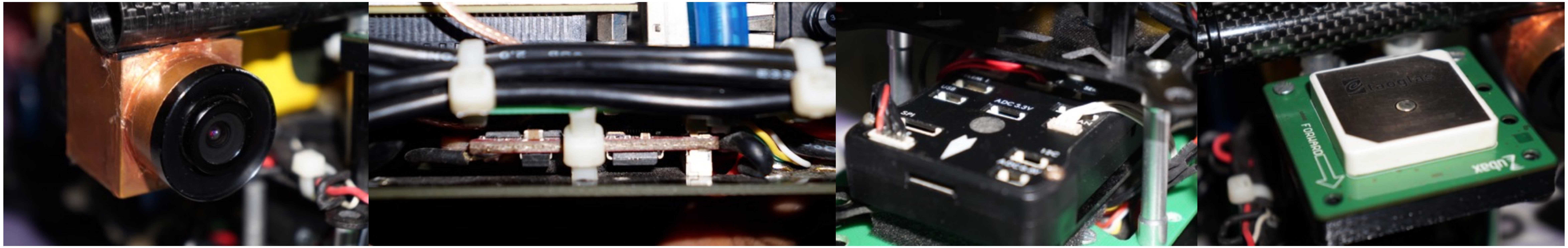
CHALLENGES

- Multicopters are highly dynamic systems - they are inherently unstable and require active control strategies for stable flight.
- System dynamics are coupled and fast.
- Limited in terms of onboard computing and sensing hardware that can be carried.
- Careful algorithm planning and time investment is required to take full advantage of GPU computing.

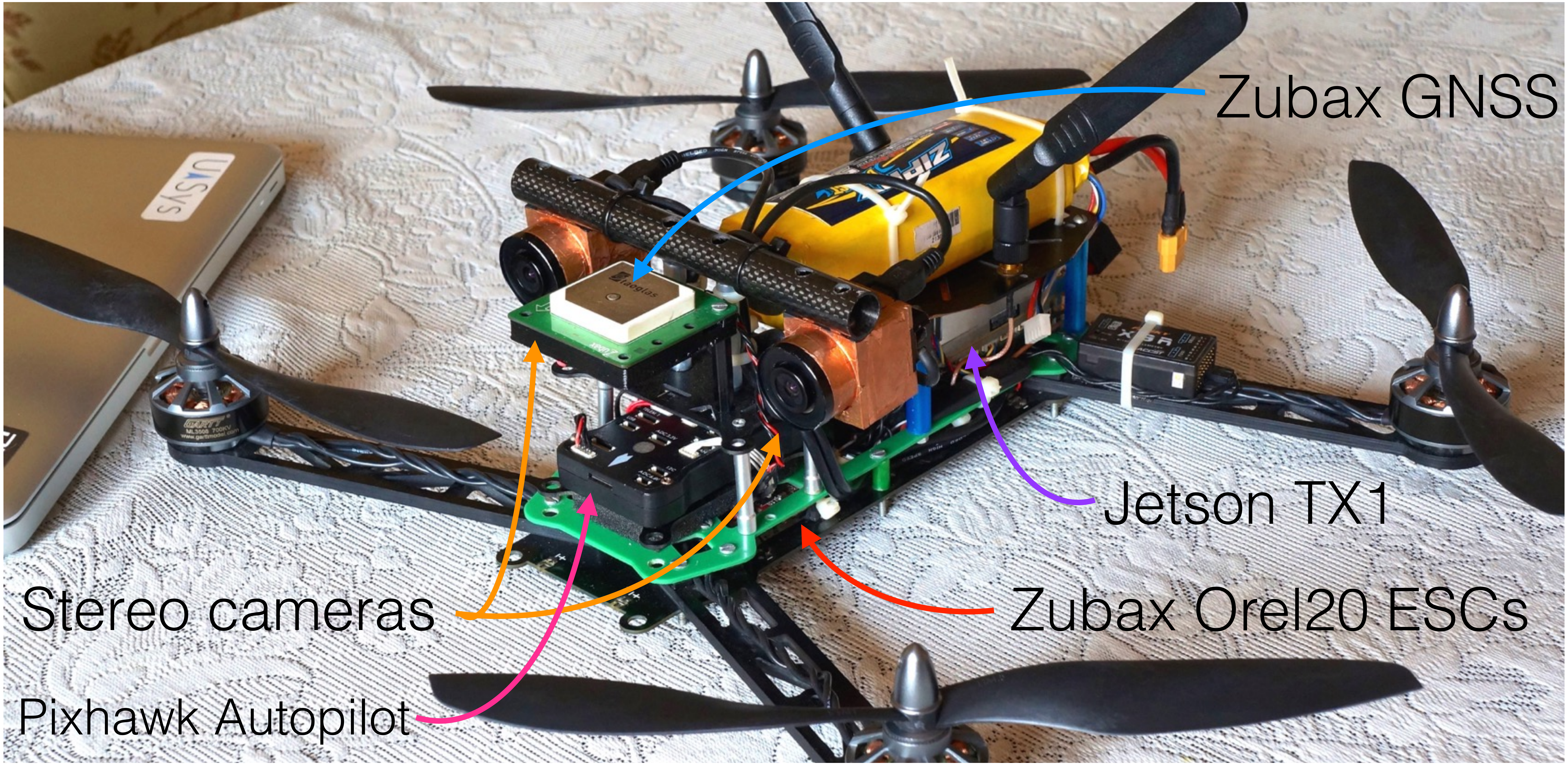


DESIGN GOALS

- Be capable of all functions of the previous generation Artemis vehicle (GPS-denied state estimation, obstacle avoidance with stereo)
- Be small enough to fly inside tight spaces like storm drains and boilers to allow use in real world situations.
- Realtime high speed reactive obstacle avoidance.
- Seamless indoor-outdoor transition capability.
- Increase algorithm efficiency and performance by taking advantage of the GPU wherever possible.



VEHICLE PLATFORM - Artemis MAV5



Zubax GNSS

Jetson TX1

Stereo cameras

Zubax Orel20 ESCs

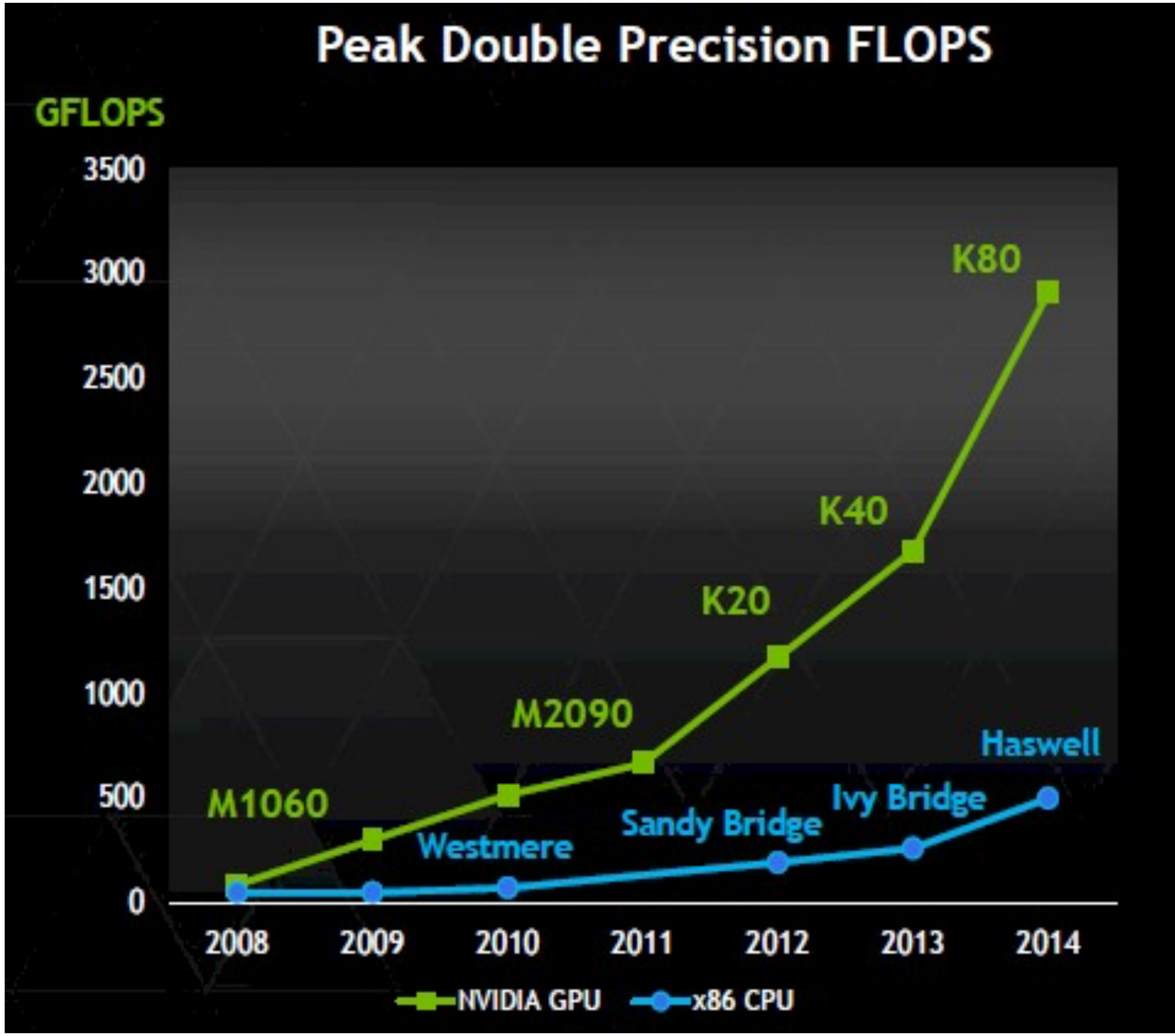
Pixhawk Autopilot

VEHICLE SPECS

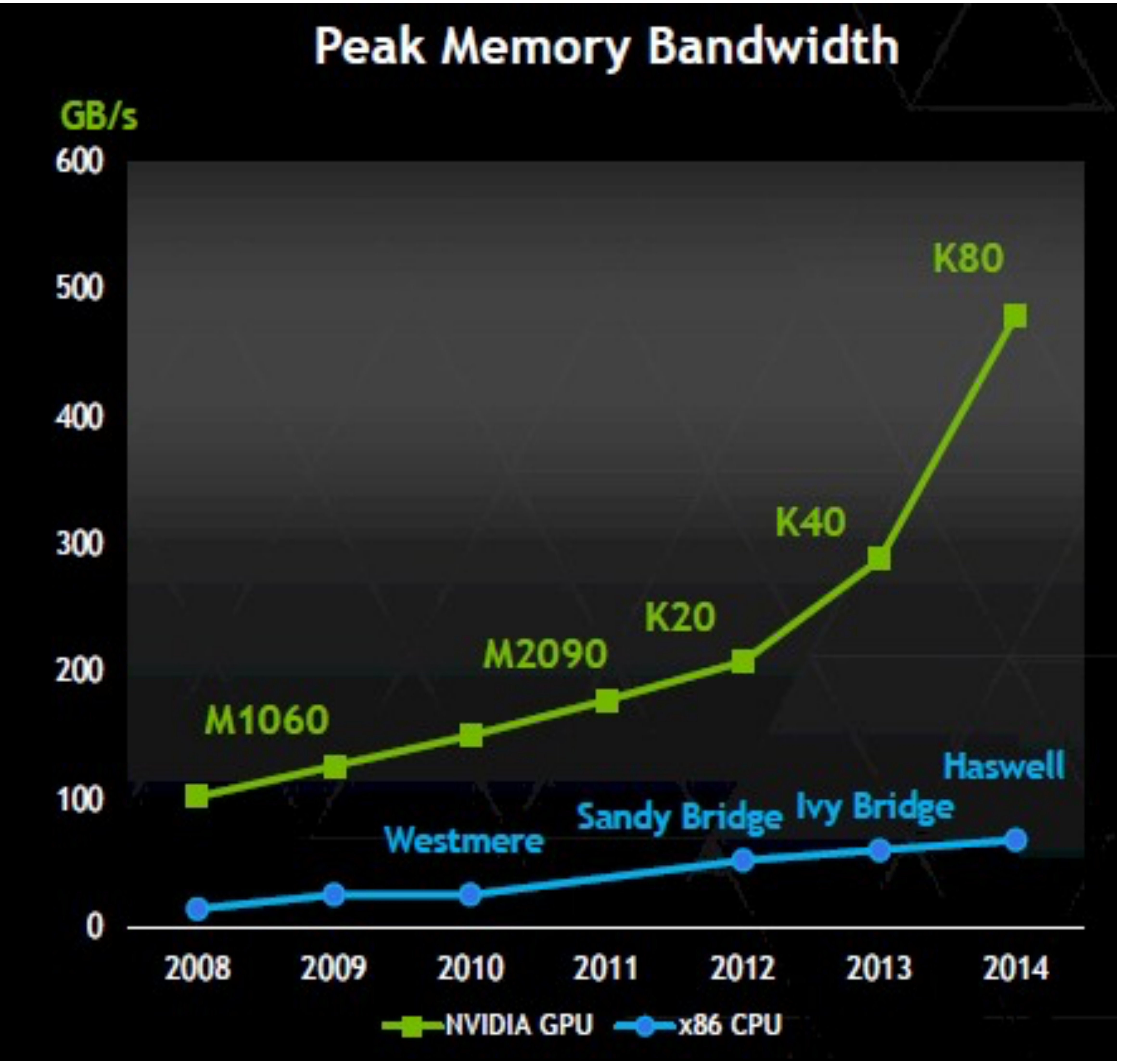
- The current-generation developmental prototype was designed after multiple iterations, building on top of our previous visual MAVs.
- Jetson Tegra X1 onboard computer running Ubuntu 14.04 (JetPack 23.1).
- Auvidea J120 carrier board exposing the I/O from the Tegra SoM.
- Pixhawk autopilot running the PX4 Flight-stack.
- Ubiquiti Rocket M5 long-range wireless datalink.
- 2 x IDS uEye SE cameras in synchronised stereo configuration, interfaced via USB 2.0
- Zubax GNSS v2 interfaced via CAN bus.
- Zubax Orel 20 motor controllers interfaced via the CAN bus.
- 20 minute flight time.

PROJECT ARTEMIS

WHY GPUs?



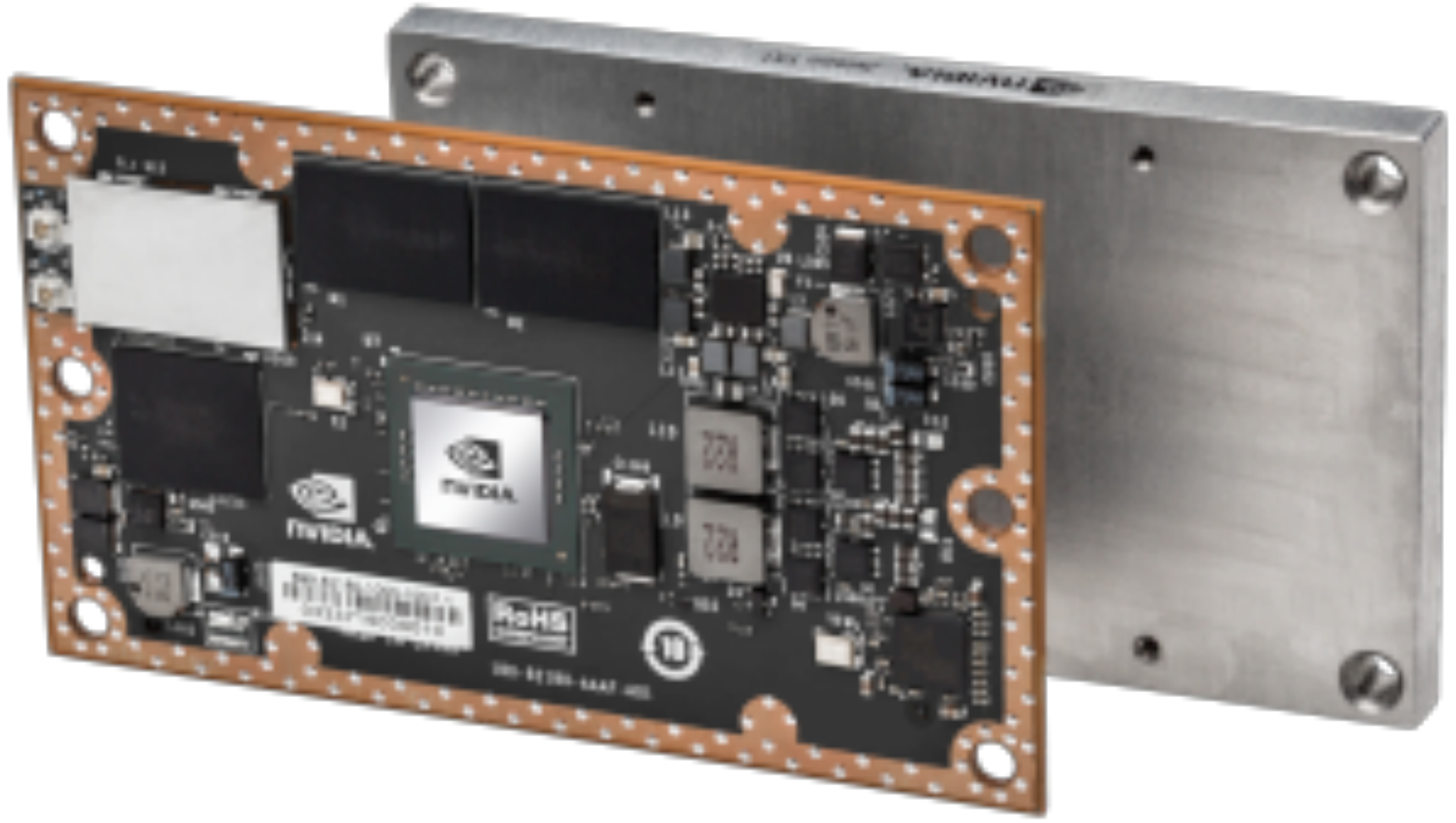
Peak Double Precision FLOPS



Peak Memory Bandwidth

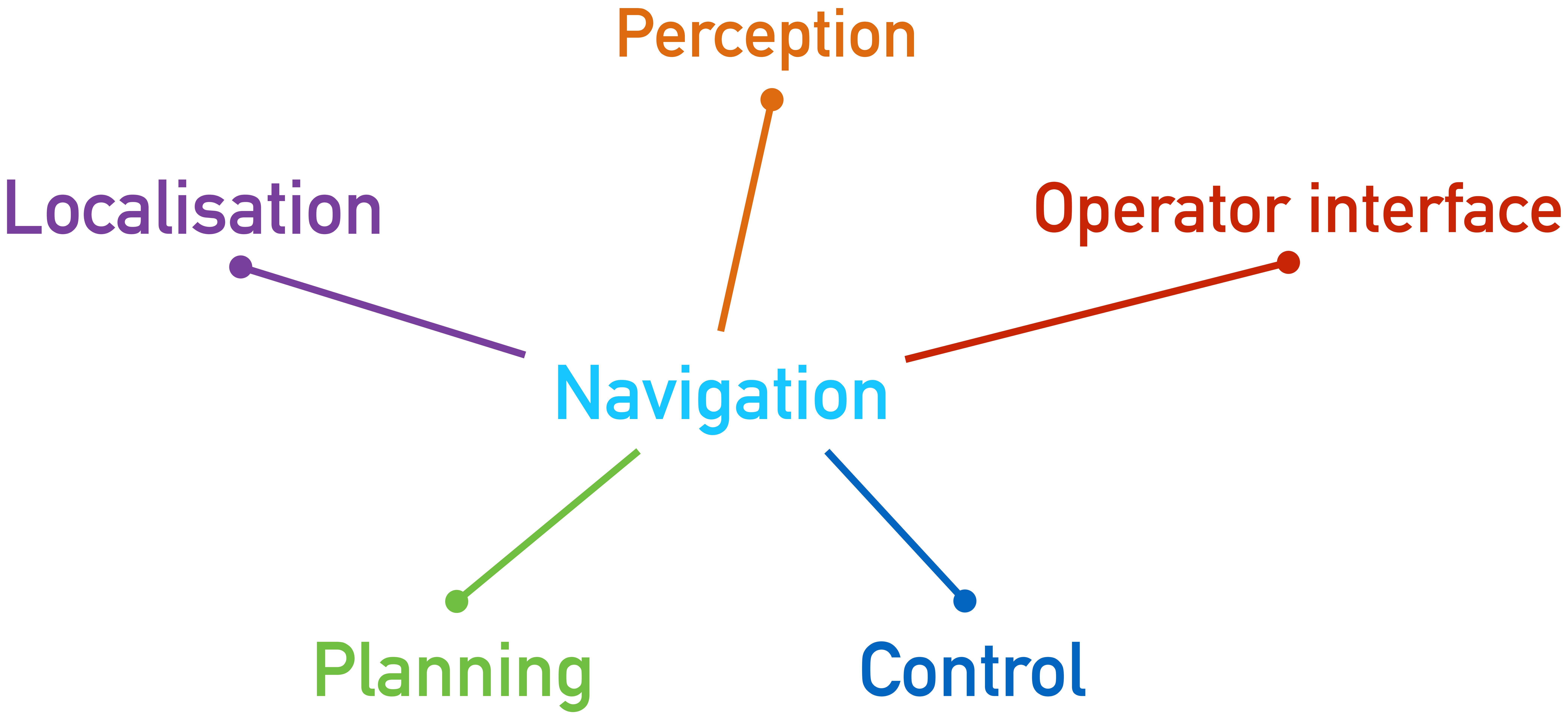
TEGRA X1 COMPUTE PLATFORM

- 1024-GFLOP Maxwell GPU.
- 64-bit quad-core ARM Cortex-A57 CPU.
- Comes as a credit-card sized SoM (50x87mm.)
- Suitable for deployment onboard embedded systems with constrained size, weight, and power (SWaP.)



- Outperforms the Core i7-6700K Skylake series processor in terms of perf-per-watt!
- Low power consumption (10W TDP.)
- Supports CUDA (Compute Unified Device Architecture) parallel programming API for fast algorithm implementation on the GPU.

THE NAVIGATION PROBLEM



Perception

Localisation

Operator interface

Navigation

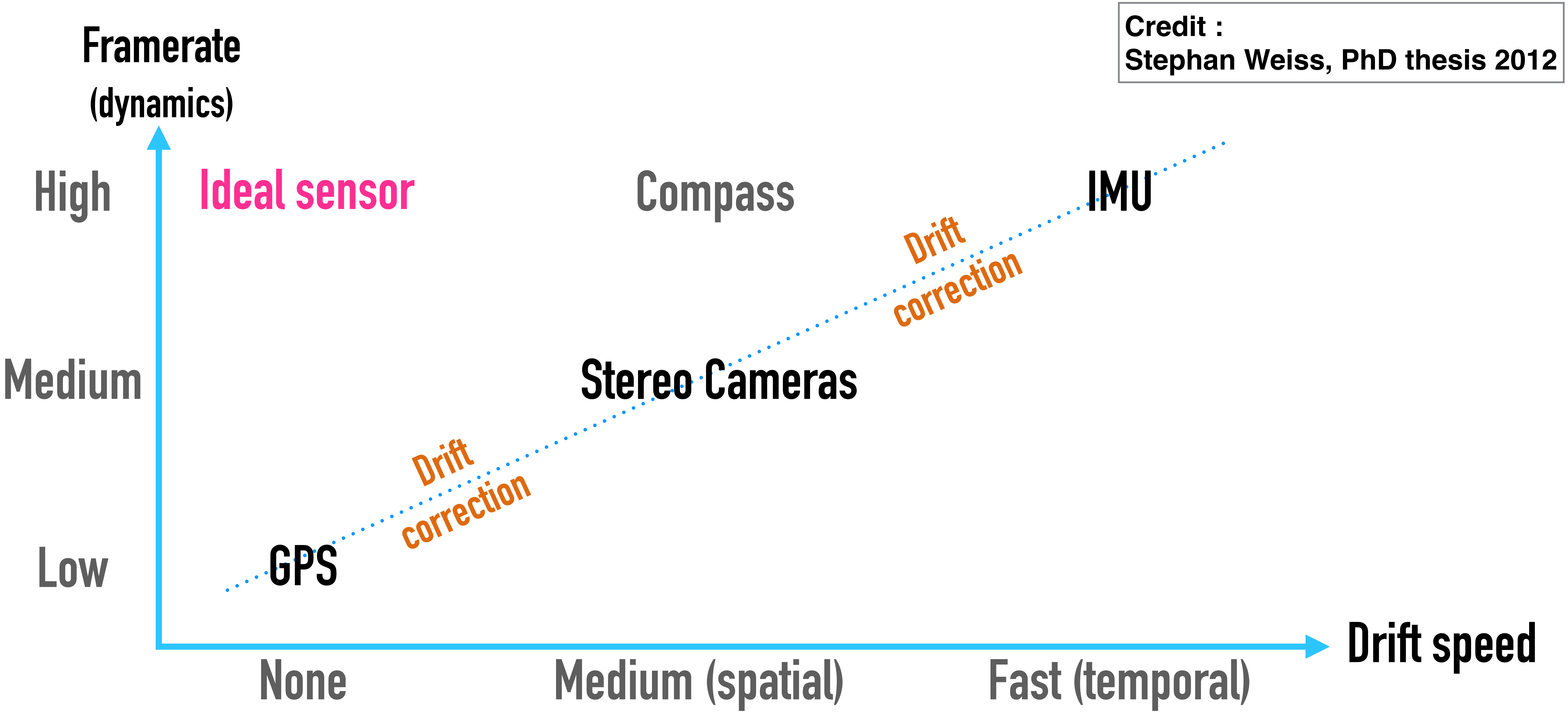
Planning

Control

PROJECT ARTEMIS

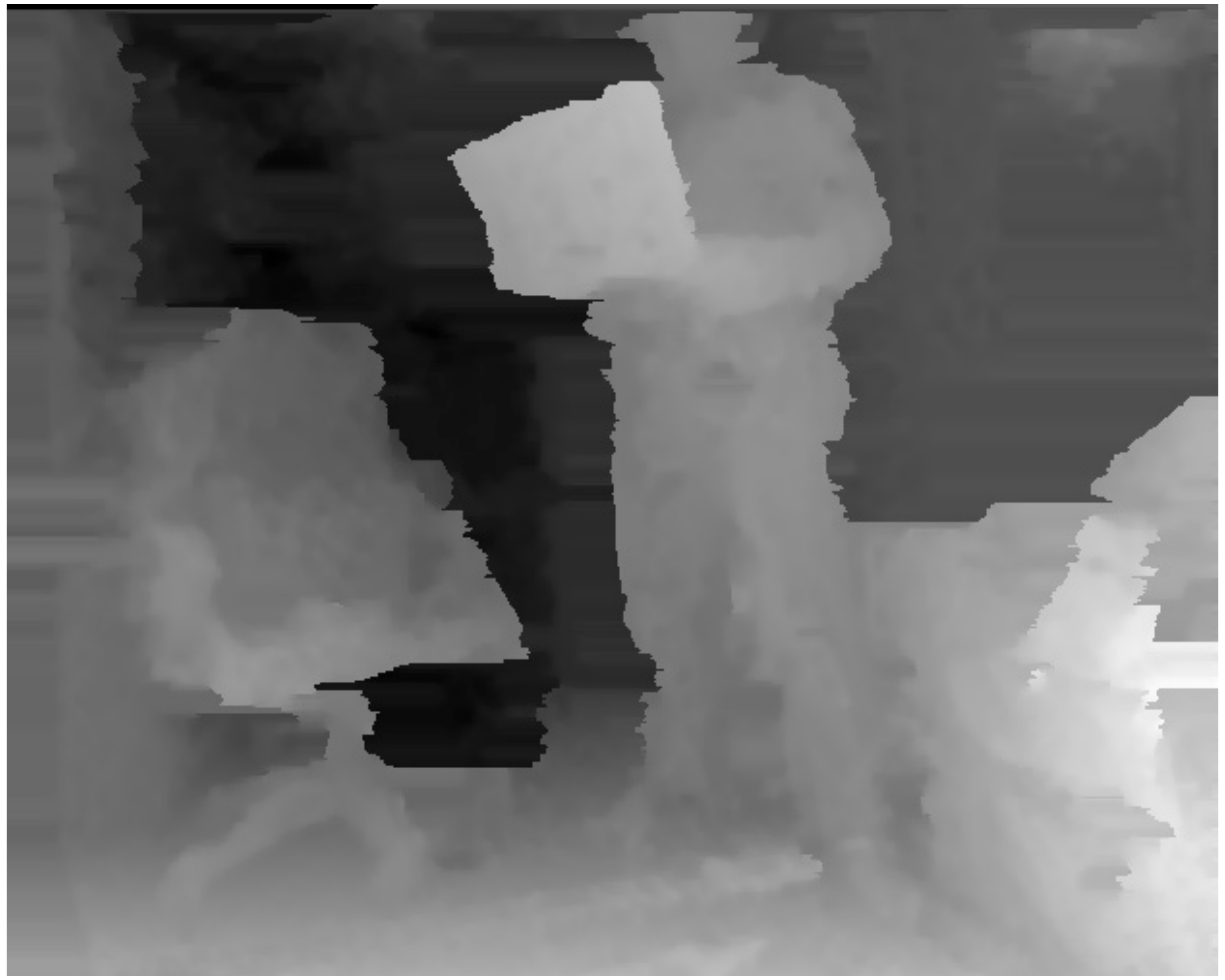
SENSING SUITE

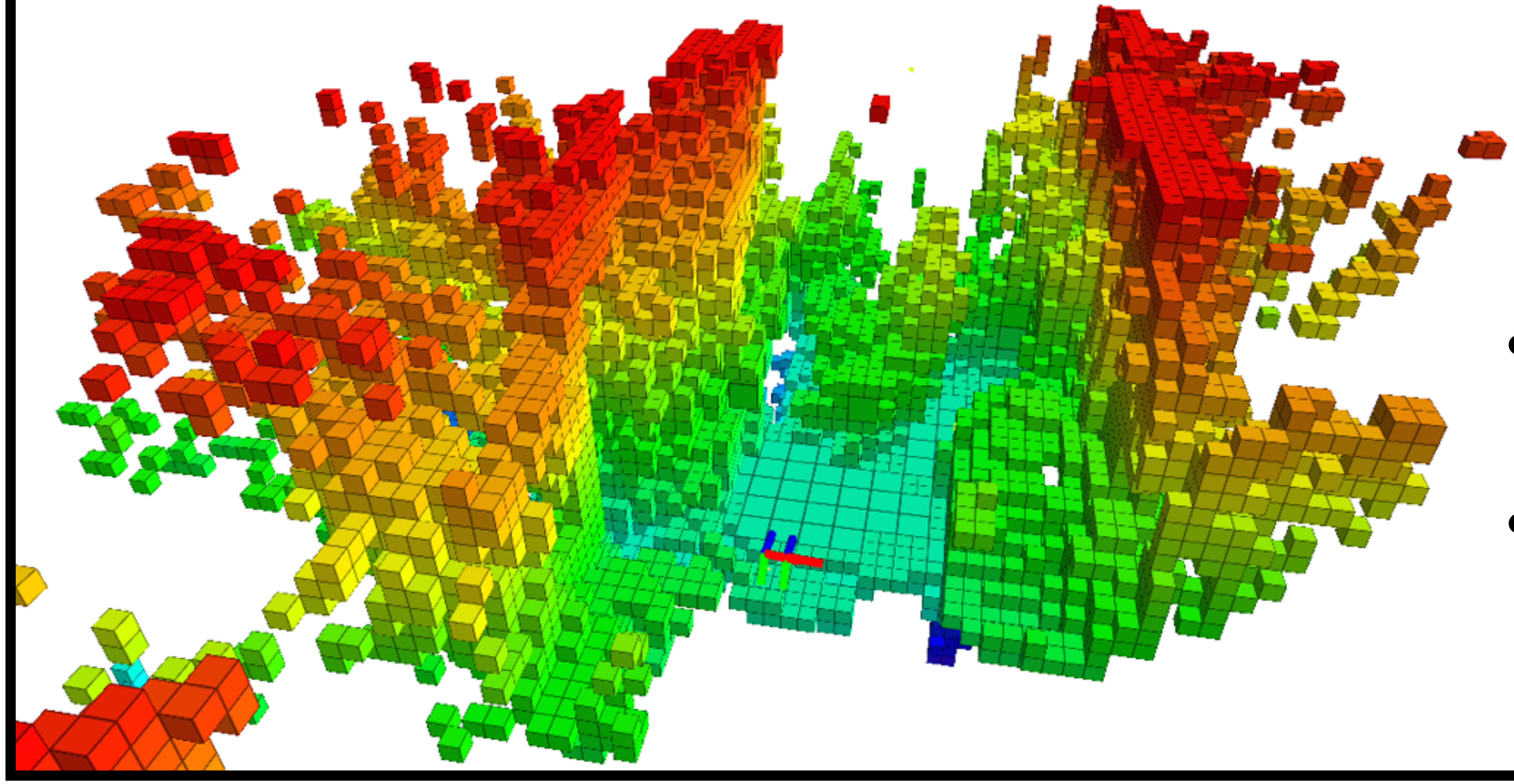
Credit :
Stephan Weiss, PhD thesis 2012



PERCEPTION

- Forward stereo cameras provide exteroceptive information about the environment.
- The cameras and the IMU are synchronised in time with respect to each other.

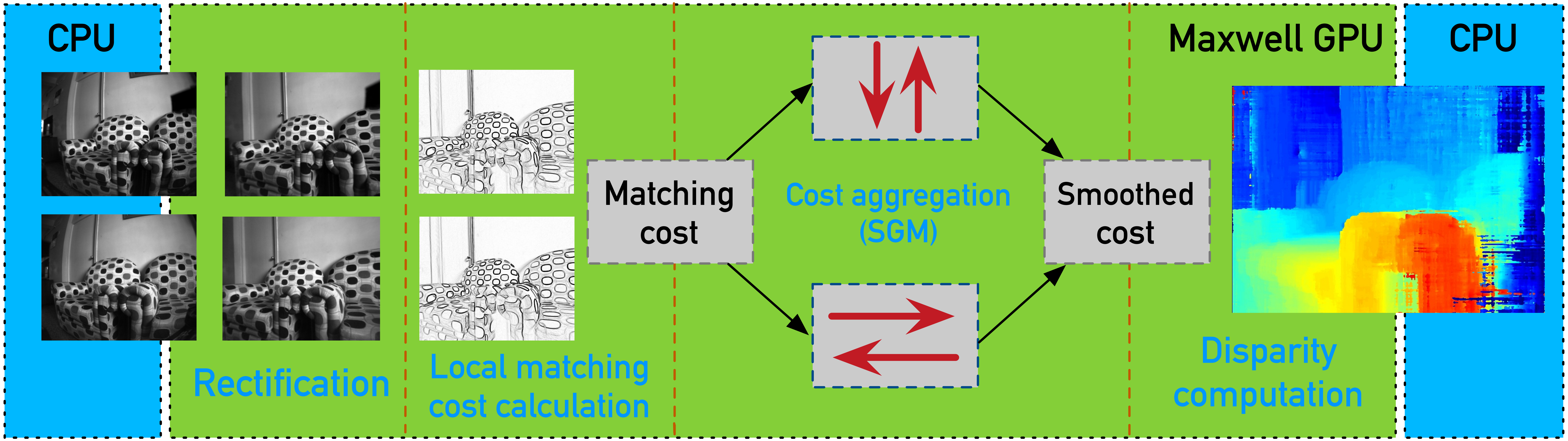




- Stereo pairs are used to compute depth images on the GPU
- Depth images are used to build a local map of the environment.

GPU ACCELERATED STEREO [1]

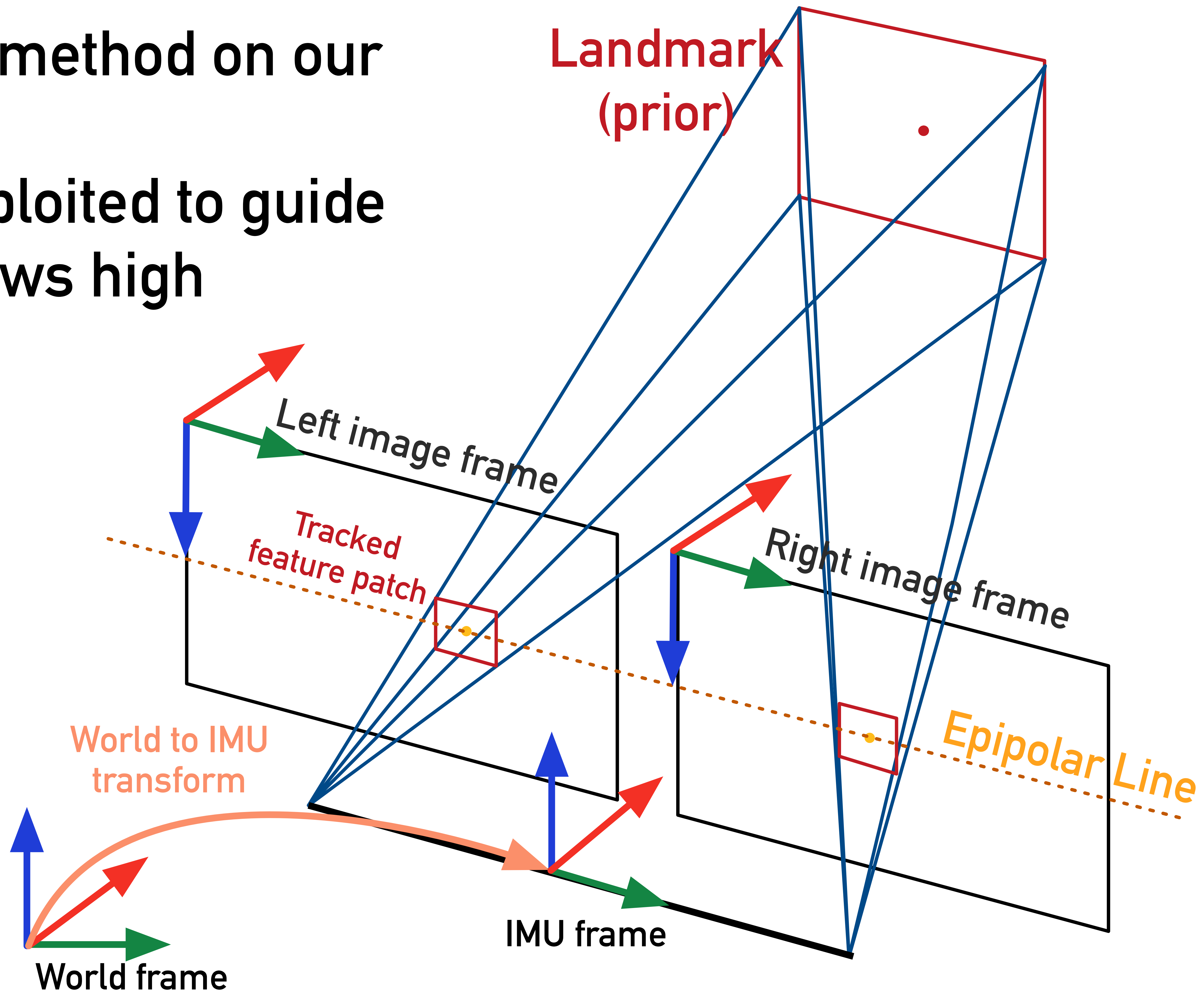
- The stereo vision pipeline is completely implemented on the GPU, exploiting a massively parallelised version of the SGM (Semi-Global Matching) algorithm. Stereo depth estimation runs at 30Hz.



[1] Based on Embedded real-time stereo estimation via Semi-Global Matching on the GPU, D. Hernandez-Juarez et al, ICCS 2016.

VISUAL-INERTIAL ODOMETRY [1]

- We use a VIO (Visual-Inertial Odometry) method on our robot for pose estimation.
- Tight IMU-camera synchronisation is exploited to guide image-space measurements, which allows high bandwidth estimation.
- Landmarks in the environment are directly tracked as filter states in an EKF (Extended Kalman Filter) and allows us to recover the full robot pose.
- Fully robotcentric feature parameterisation allows for a truly power-up-and-go system, with no extra initialisation step required.

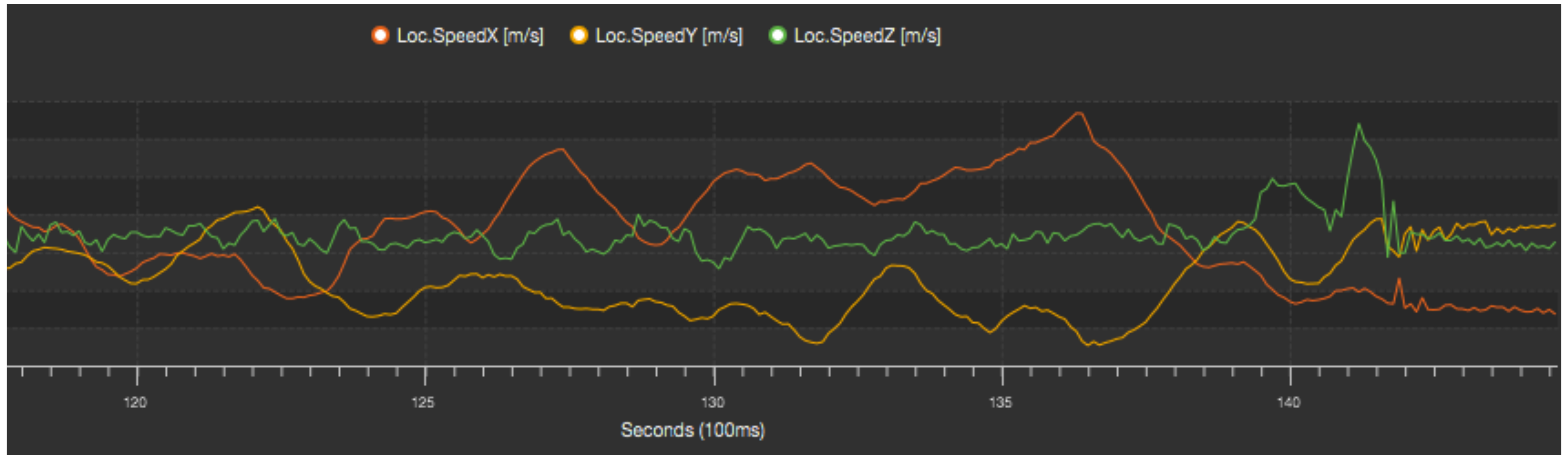


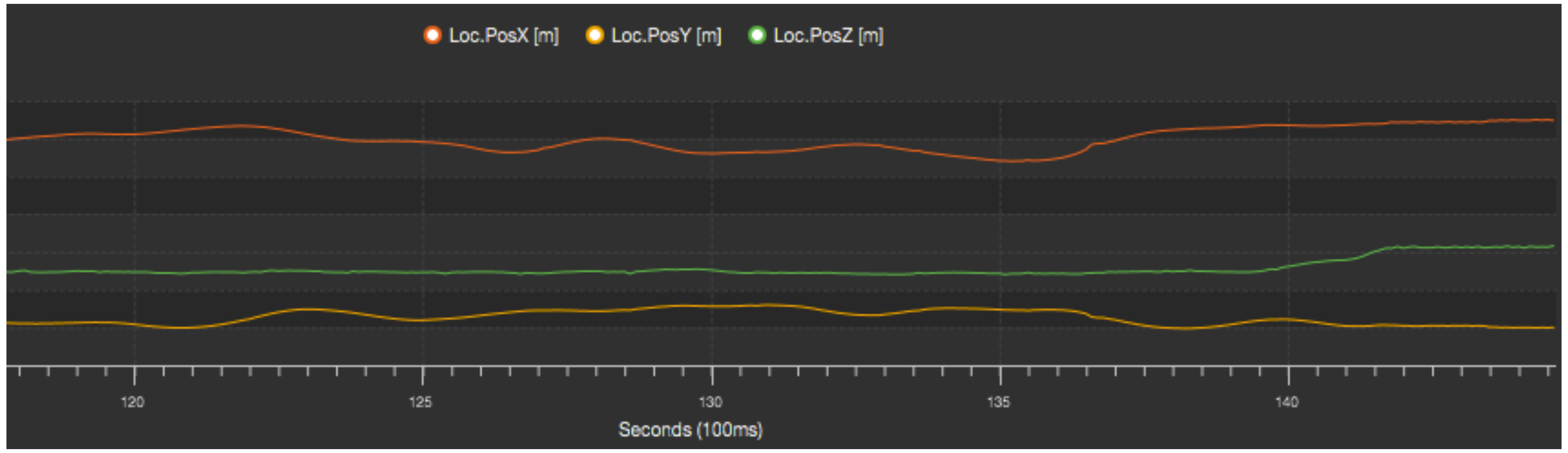
[1] Based on Robust Visual Inertial Odometry Using a Direct EKF-Based Approach, Michael Bloesch et al, IROS 2015.

STATE ESTIMATION

$$x_f = [\mathbf{p}_w^{i^T}, \mathbf{v}_w^{i^T}, q_w^{i^T}, \mathbf{b}_\omega^{i^T}, \mathbf{b}_a^{i^T}]$$

- The system localises itself using a combination of GPS and vision.
- Inertial measurements are used to propagate the state of the filter, while the visual information is taken into account during the filter update steps. GPS observations are fused when sufficiently accurate.
- Fusing these complementary data sources at the lowest possible level allows the estimated system state to constrain and guide the image-space measurements, enforcing consistency between image-space feature positions and 6DOF vehicle motion.

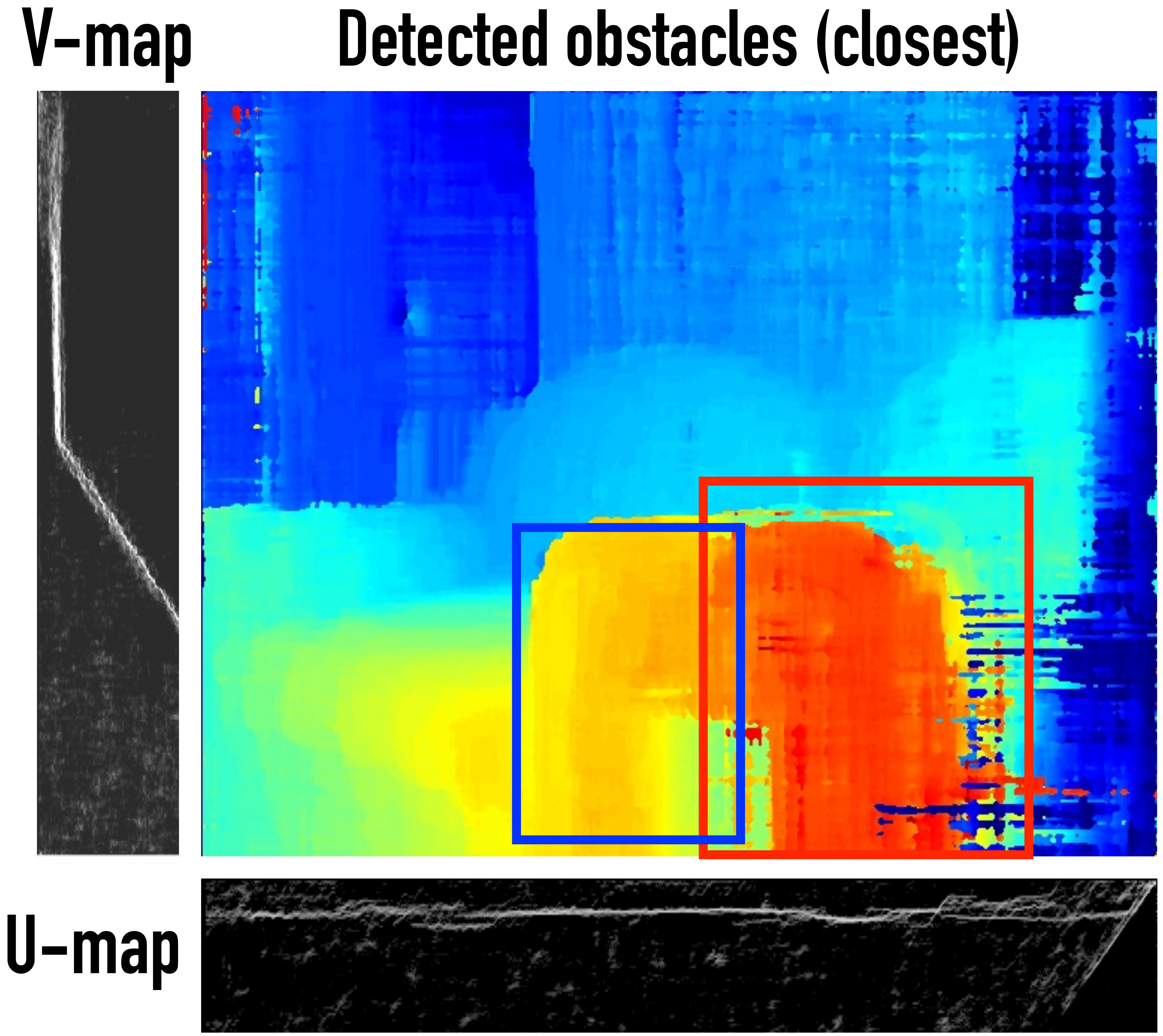




MULTI-SENSOR STATE ESTIMATION

REACTIVE OBSTACLE AVOIDANCE

- Low latency detection and reaction is necessary for successful high speed avoidance.
- Total time budget from acquisition to outputting a new waypoint is only 30 ms.
- We use ‘U-maps’[1] to efficiently segment obstacles directly from disparity data.
- Our avoidance algorithm works directly on the disparity data, and plans in local map space. No global map required.

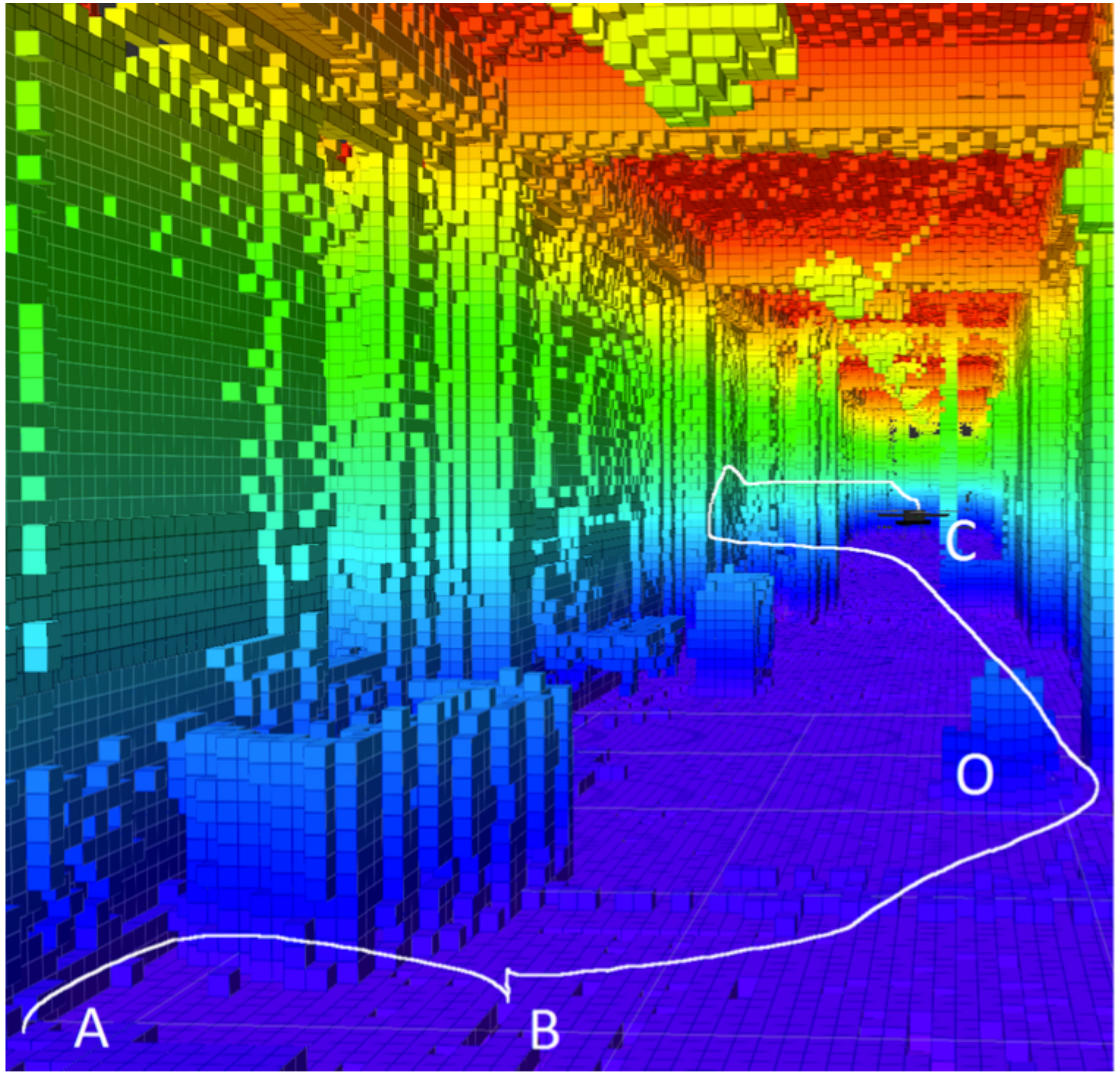


[1] R. Labayrade, D. Aubert, and J.-P. Tarel, “Real time obstacle detection in stereovision on non flat road geometry through ”v-disparity” representation,” in Intelligent Vehicle Symposium, 2002. IEEE, vol. 2, pp. 646–651, IEEE, 2002.

OBSTACLE DETECTION

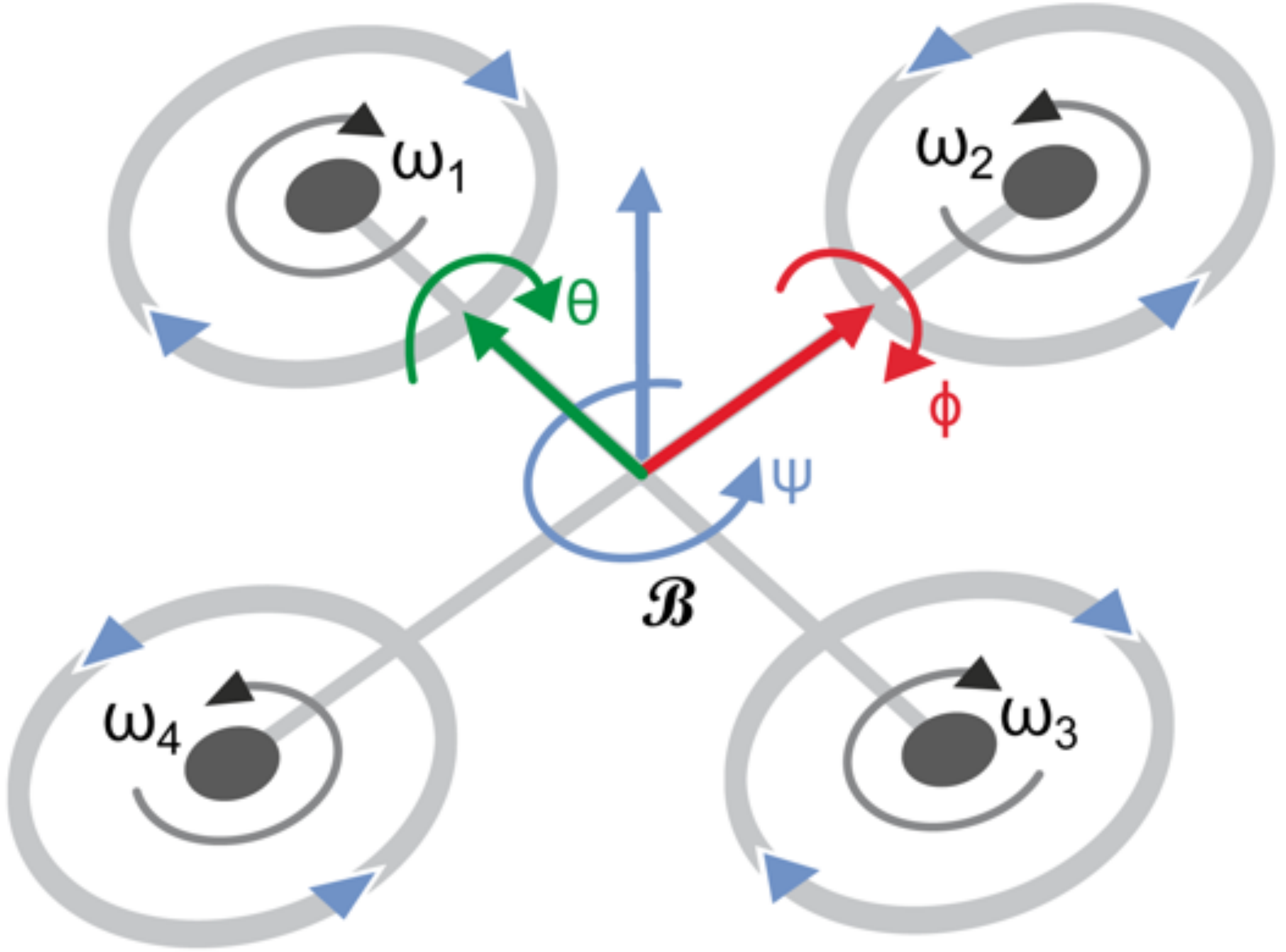
GLOBAL PLANNING

- Currently a work-in-progress. Traditional approaches are too resource heavy for this embedded system.
- We plan to exploit the RRT (Rapidly Expanding Random Trees) family of algorithms which can be massively parallelised to run efficiently on the GPU.
- We need a faster volumetric mapping approach than Octomaps. Currently a GPU-space solution is being experimented with.
- Autonomous exploration approach needs to be re-implemented to take advantage of GPU-accelerated RRT.



NONLINEAR OPTIMAL CONTROL

- We developed a fully nonlinear model predictive controller for our quadrotor, which allows accurate trajectory following.
- Our controller receives high level paths from a navigator module, and then locally deforms these paths continuously using obstacle data.
- Model predictive control integrates the planning of the optimal trajectory with state feedback in the control loop.
- The system can thus react to sudden, unpredictable changes in the environment, as is required in dynamic environments.



OPERATOR INTERFACE

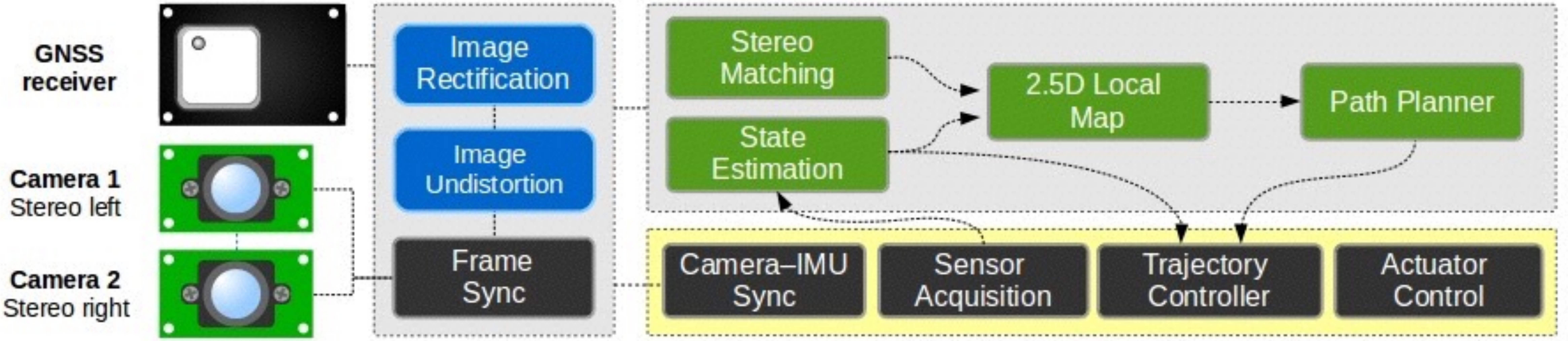
- We use a single tablet and R/C controller as our operator control system.
- The tablet displays a live FPV (first-person-view) feed. The operator can use this feed to fly.
- We adopted the open-source multi platform ground control software QGroundControl for our system.
- The front view of the vehicle is visualised as 4 quadrants, with quadrants lighting up as obstacles are detected.
- Beeps signify the distance to the closest obstacle. The vehicle automatically brakes or applies path corrections if the obstacle is too close.

SOFTWARE FRAMEWORK

- Software architecture follows a high-level / low-level separation model for maximal reliability. The flight core is isolated from the application-level processing to ensure stability of core vehicle operation, independent of the high-level system state.
- Low-level tasks critical to flight control, like motor actuation and attitude estimation run on the PX4 Middleware on the NuttX RTOS.
- High-level tasks like computer vision run on the TX1 computer, on top of the ROS (Robot Operating System) Middleware.

COMPLETE NAVIGATION PIPELINE

- The navigation pipeline overall operates at 30 Hz, which is the framerate of the camera system. Currently limited by the USB 2.0 cameras we used.
- State estimation runs at IMU rate, 200Hz.
- Time available from image acquisition to outputting new position/velocity commands is only about 30ms.
- High level and low level task split allows the TX1 computer and autopilot to maintain low-latency control performance.



AUTONOMOUS FLIGHT

THANK YOU!

QUESTIONS?

Scan me!

Scan me!

Our open-source software stack is available at
www.github.com/ProjectArtemis

www.uasys.io/research

Contact me at kabir@uasys.io for any licensing queries, systems integration services or if you'd just like to chat!