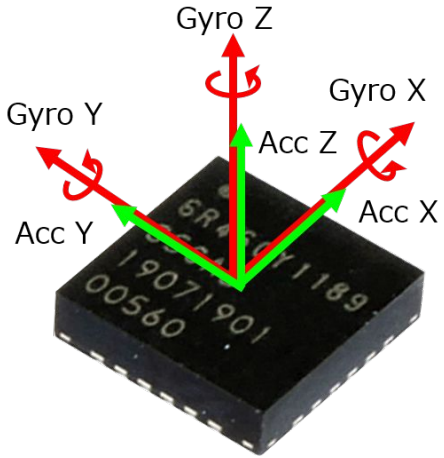
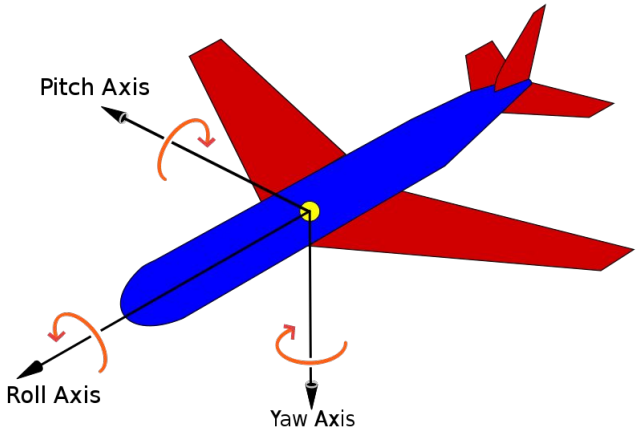
IMU-Camera Sensor Fusion

Chengyu Liu & Hanbo Wang

Introduction

- Inertial Measurement Unit
- Typically consists of accelerometer and gyroscope (6-DOF)
- Fuse the data from the two sensors to obtain accurate attitude estimation (roll, pitch, yaw)
- Applications include drones, smartphones, and virtual reality systems

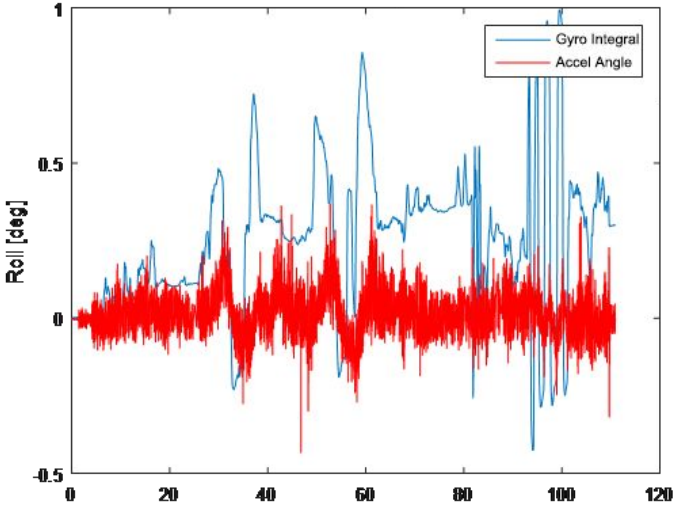




Purpose of Sensor Fusion

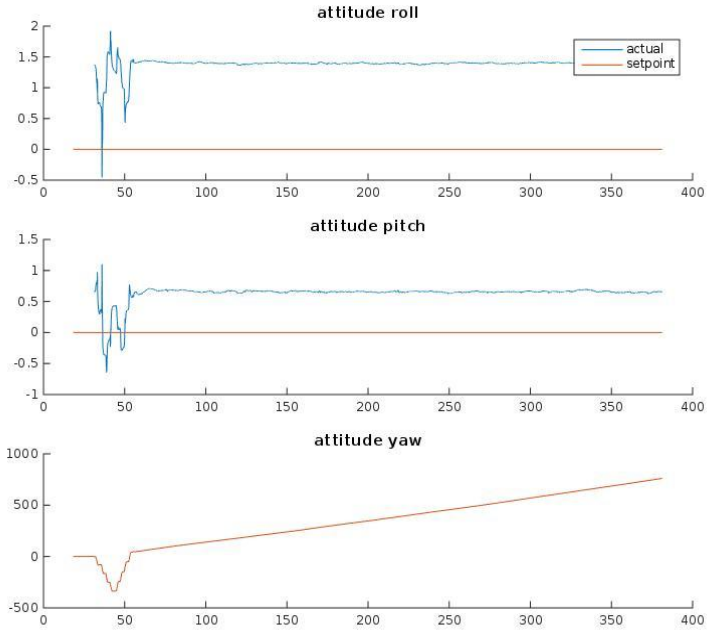
- Accelerometers are very sensitive to vibration, but readings are reliable after the filtering algorithm when operated for a long time
- Gyroscopes are not sensitive to vibration but the orientation will drift over time due to repeated integration of error

- Example of roll angle calculated by accel or gyro alone
  - Gyroscope: $\Theta = \int \omega_z(t)dt$
  - Accelerometer: $\Theta = \text{atan2}(a_x, \sqrt{(a_y^2+a_z^2)})$



Yaw Drift

- Caused by a variety of reasons
  - biases in the sensors
  - temperature changes
  - accumulation of errors over time
- Pitch and roll do not typically suffer from drift
- Conventional solution: magnetometer (compass)
- However, the sensors may fail indoors due to ferrous structural members

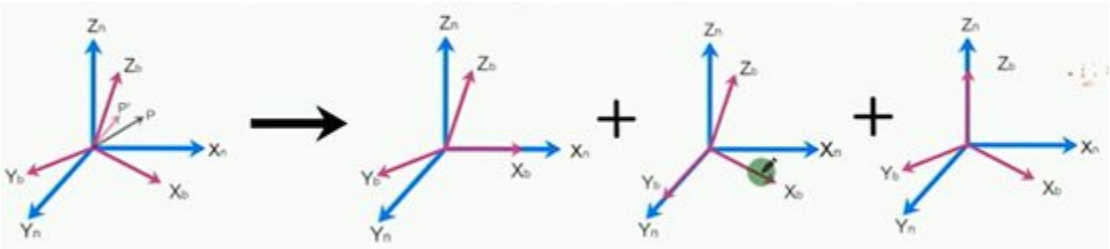


Yaw Drift

- Our solution: camera calibration (visual compass)
  - Install a camera in front of the drone
  - If the yaw angle of the drone is rotated clockwise by 8 degrees, the corresponding camera image will shift some pixels
  - Combining the data from IMU and camera

Fusion Algorithm

- Direction Cosine Matrix - DCM
  - A mathematical representation of the orientation of a body in three-dimensional space
  - Can be used to convert a vector from one coordinate system to another



Fusion Algorithm

- Direction Cosine Matrix - DCM
  - Rotate around X-axis

$$Ax = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & sin(\alpha) \\ 0 & -sin(\alpha) & cos(\alpha) \end{bmatrix}$$

  - Rotate around Y-axis

$$Ay = \begin{bmatrix} cos(\alpha) & 0 & -sin(\alpha) \\ 0 & 1 & 0 \\ sin(\alpha) & 0 & cos(\alpha) \end{bmatrix}$$

  - Rotate around Z-axis

$$Az = \begin{bmatrix} cos(\alpha) & sin(\alpha) & 0 \\ -sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  - A=Ax*Ay*Az

A=

$$\begin{bmatrix} \cos\gamma\cos\psi + \sin\gamma\sin\theta\sin\psi & \cos\gamma\sin\psi + \sin\gamma\sin\theta\cos\psi & -\sin\gamma\cos\theta \\ -\cos\theta\sin\psi & \cos\theta\cos\psi & \sin\theta \\ \sin\gamma\cos\psi + \cos\gamma\sin\theta\sin\psi & \sin\gamma\sin\psi - \cos\gamma\sin\theta\cos\psi & \cos\gamma\cos\theta \end{bmatrix}$$

Fusion Algorithm

- Limitations of Direction Cosine Matrix - DCM
  - It is a computationally intensive process. This can be a problem in applications where real-time performance is important
  - DCM has Gimbal lock problem. This can occur when the body's pitch angle is 90 degrees, which can result in a loss of one degree of freedom
- Solution: using quaternion representation

$$\begin{cases} q_0 = \cos\frac{\theta}{2} \\ q_1 = l\sin\frac{\theta}{2} \\ q_2 = m\sin\frac{\theta}{2} \\ q_3 = n\sin\frac{\theta}{2} \end{cases} \Rightarrow C_b^R = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_3q_2-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_3q_2+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix} \Rightarrow \begin{cases} \theta = arcsin[2(-q0q1+q2q3)] \\ \gamma = -arctan(\frac{q0q2+q1q3}{q0^2-q1^2-q2^2+q3^2}) \\ \psi = -arctan(\frac{q1q2+q0q3}{q0^2-q1^2+q2^2-q3^2}) \end{cases}$$

Fusion Algorithm

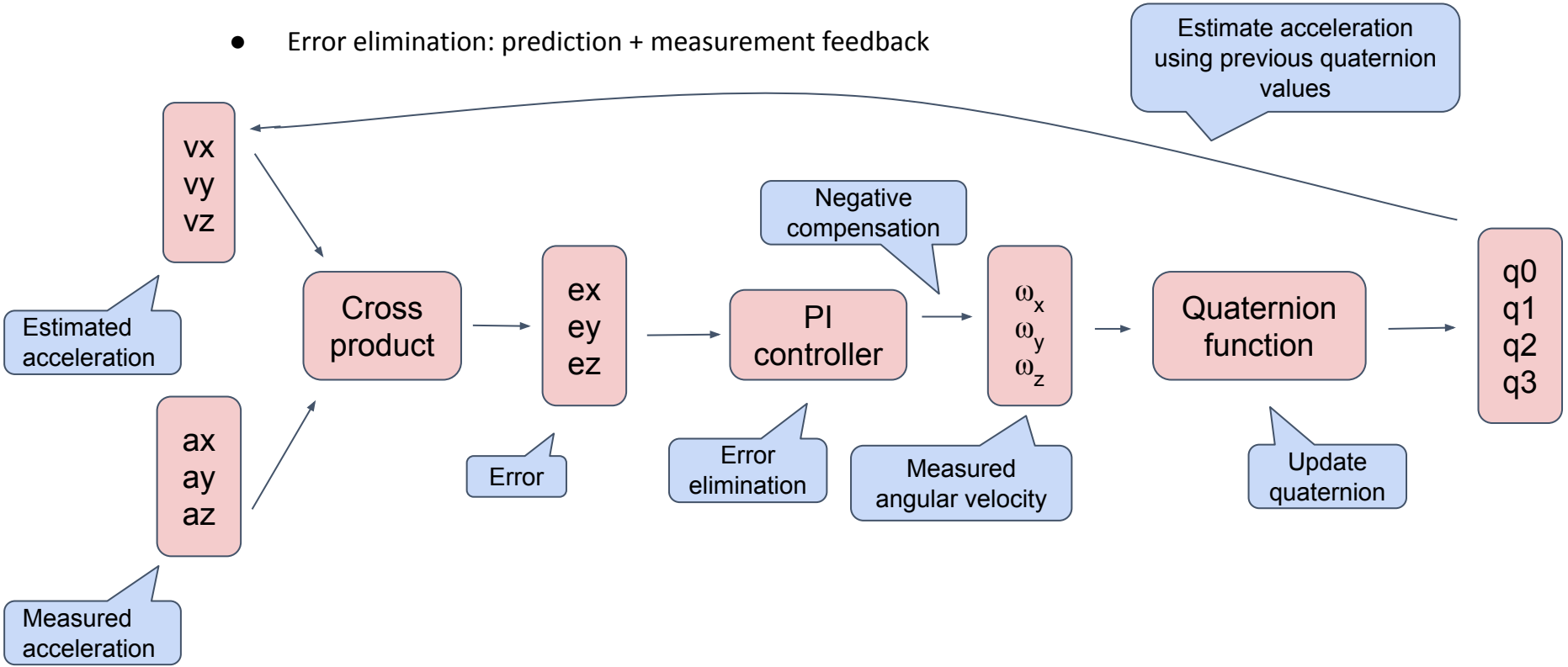
- Differentiating quaternions with respect to time and then solve the differential equation
- Update quaternion using the following equation

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}_{t+\Delta t} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}_t + \frac{\Delta t}{2} \begin{bmatrix} -\omega_x q_1 - \omega_y q_2 - \omega_z q_3 \\ \omega_x q_0 + \omega_z q_2 - \omega_y q_3 \\ \omega_y q_0 - \omega_z q_1 + \omega_x q_3 \\ \omega_z q_0 + \omega_y q_1 - \omega_x q_2 \end{bmatrix}$$

where $\omega_x, \omega_y, \omega_z$ denote the angular velocity components along each axis

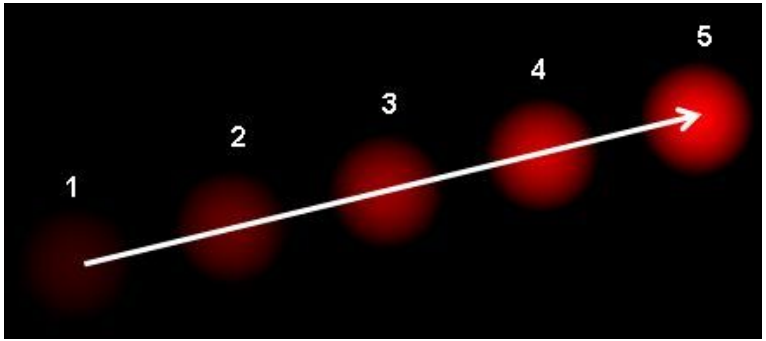
Fusion Algorithm

- Error elimination: prediction + measurement feedback



Optical Flow

Optical flow is the pattern of apparent motion of image objects between two consecutive frames caused by the movement of object or camera.



Assumptions

- Constant brightness: The pixel intensities of an object do not change between consecutive frames
- Time Continuity: The change in time does not cause a drastic change in the pixel position so that the gray value of the pixel can find the corresponding partial derivative of the position
- Spatial Consistency: Neighbouring pixels have similar motion

Optical Flow

$$I(x, y, t) = I(x + dx, y + dy, t + dt)$$

$I(x,y,t)$ is the intensity of one pixel in first frame

$I(x+dx,y+dy,t+dt)$ is the intensity of same pixel in second frame

$$I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t$$

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t} = 0$$

$$\frac{\partial I}{\partial x}\mu + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

μ and v are the velocity components along x and y direction

Lucas-Kanade method

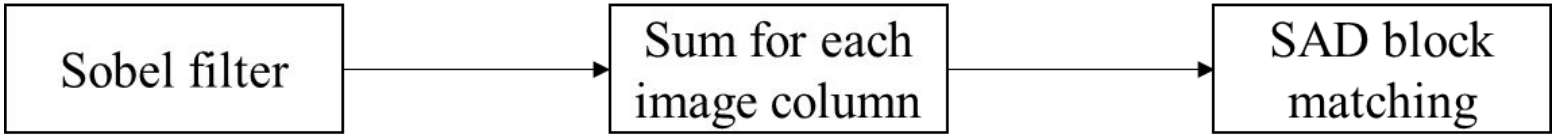
LK method assumes that the flow is essentially constant in a local neighbourhood of the pixel under consideration, and solves the basic optical flow equations for all the pixels in that neighbourhood, by the least squares criterion.

$$
\begin{aligned}
I_{x1}u + I_{y1}v &= -I_{t1} \\
I_{x2}u + I_{y2}v &= -I_{t2} \\
&\vdots \\
I_{xn}u + I_{yn}v &= -I_{tn}
\end{aligned}
$$

$$
\begin{bmatrix} u \\ V \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} I_{xi}^2 & \sum_{i=1}^{n} I_{xi} I_{\mathrm{yi}} \\ \sum_{i=1}^{n} I_{xi} I_{\mathrm{yi}} & \sum_{i=1}^{n} I_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{i=1}^{n} I_{xi} I_{\mathrm{ti}} \\ -\sum_{i=1}^{n} I_{yi} I_{\mathrm{ti}} \end{bmatrix}
$$

UNIVERSITY OF MICHIGAN

EdgeFlow

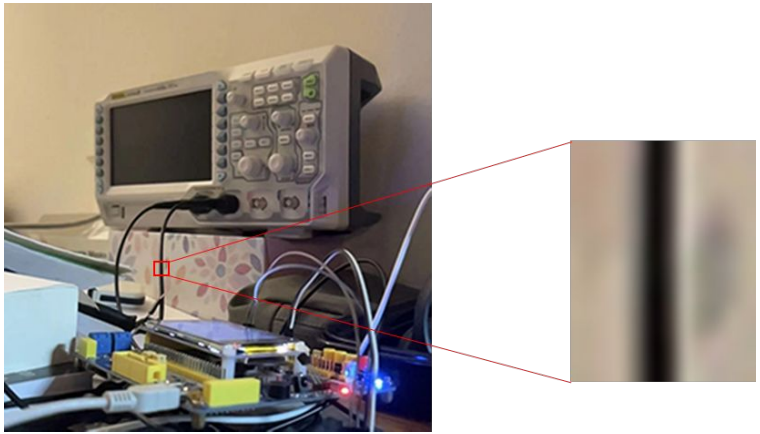


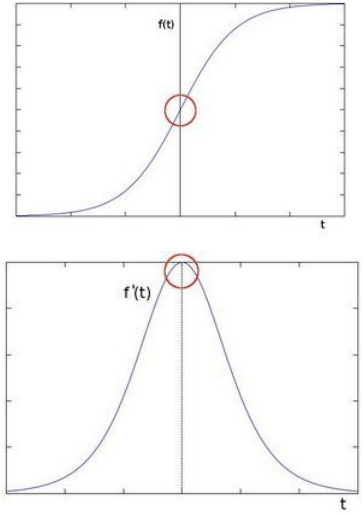
Extract image edges

reduce the 2D image search problem to 1D signal matching, increasing its computational efficiency.

From two sequential frames, these edge histograms can be calculated and matched locally with the Sum of Absolute Differences

Sobel Filter

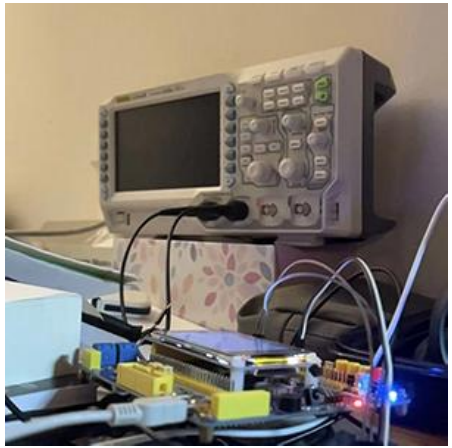




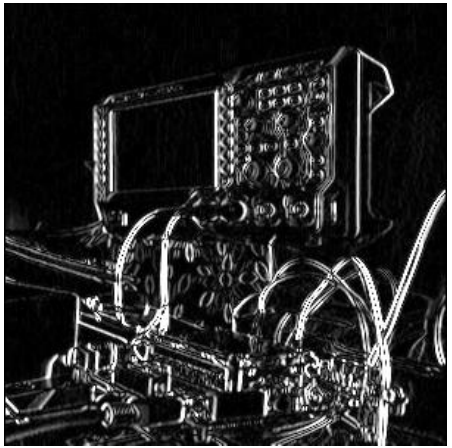
The Sobel filter is a gradient-based method that looks for strong changes in the first derivative of an image. The Sobel edge detector uses a pair of 3 × 3 convolution masks, one estimating the gradient in the x-direction and the other in the y-direction.

UNIVERSITY OF MICHIGAN

Sobel Filter



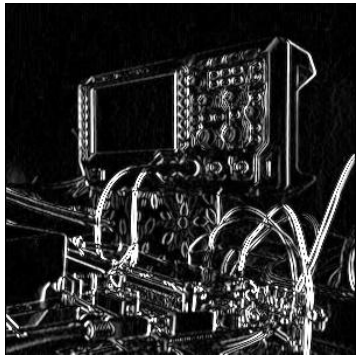
Oscilloscope image with 300*300 pixels



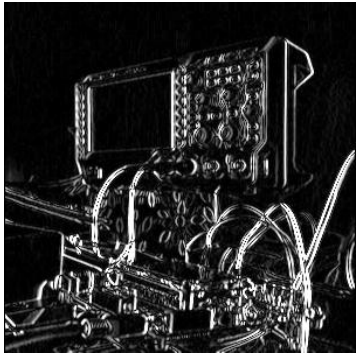
Oscilloscope image after sobel operator

Edge Features Histograms

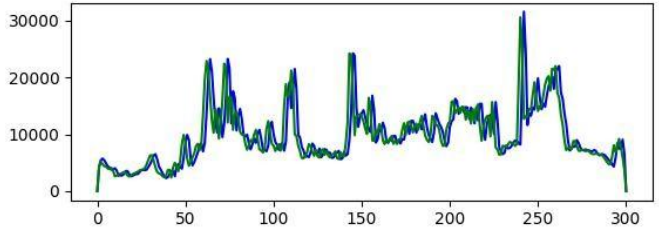
The histogram can be computed for each of the image's dimensions by summing up the intensities



Oscilloscope image after sobel operator in first frame



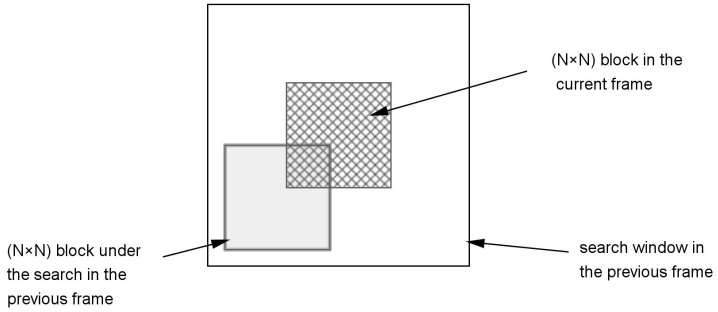
Oscilloscope image after sobel operator in next frame



The edge feature histograms of two images

SAD block matching

The sum of absolute differences (SAD) is a measure of the similarity between image blocks. It is calculated by taking the absolute difference between each pixel in the original block and the corresponding pixel in the block being used for comparison.



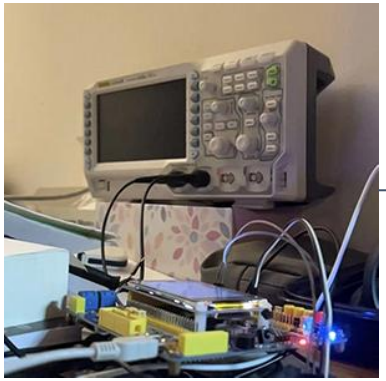
UNIVERSITY OF MICHIGAN

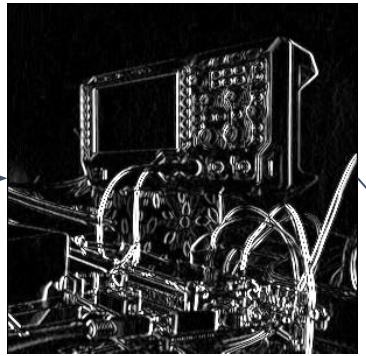
SAD block matching

- Block size
  - 16 pixels
- Maximum search distance
  - Large distance will lead to large amount of computation
  - Small distance will restrict the camera rotation angle between sequential frame
  - -10~+10 pixels
- Search strategy
  - Full search method
- Resolution
  - High resolution of image will lead to large amount of computation
  - Small resolution of image will cause small resolution of angle measurement

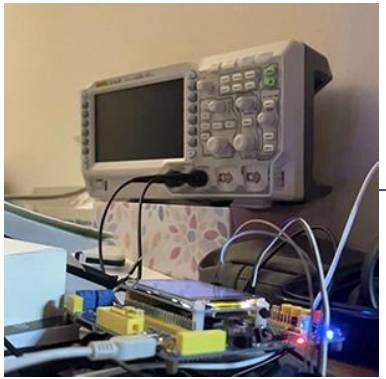
EdgeFlow Algorithm

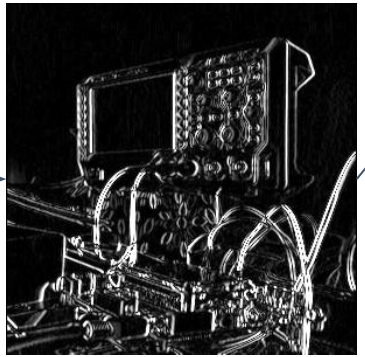
t=n-1

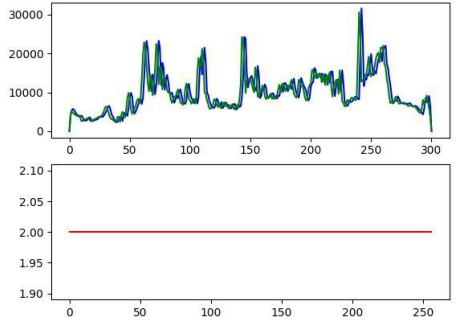




t=n

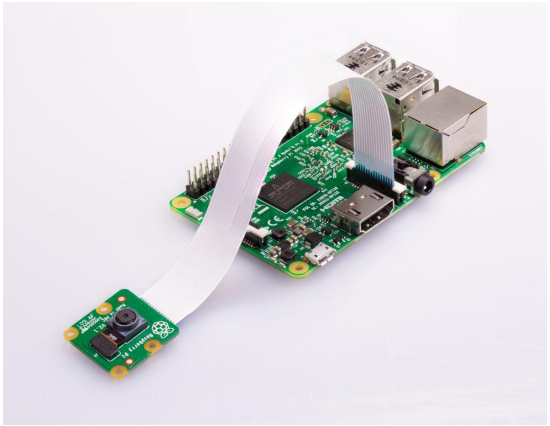






Hardwares

- MPU9250
- IMX219 Camera Module
- Raspberry Pi
- IMU and camera communicate with Raspberry Pi over IIC and MIPI interfaces



UNIVERSITY OF MICHIGAN

Performance

- IMU update rate:
  - 25 frames per second at 640*480 pixels resolution
  - 50 frames per second at 320*240 pixels resolution
  - 56 frames per second at 160*120 pixels resolution
- Yaw angle resolution:
  - 0.1° at 640*480 pixels
  - 0.2° at 320*240 pixels
  - 0.4° at 160*120 pixels
- Euler angle accuracy: 2°

UNIVERSITY OF MICHIGAN

Demo

UNIVERSITY OF MICHIGAN

Future Work

- The lightence condition of environment has great impact to the angle measurement using camera
- The movement of the object in image will influence the angle measurement using camera
- The update rate of Raspberry Pi is unstable, maybe need RTOS
- Replacing Raspberry Pi by other cheaper MCU
- More test should be done

UNIVERSITY OF MICHIGAN

References

[1] A. Assa and F. Janabi-Sharifi, "A Kalman Filter-Based Framework for Enhanced Sensor Fusion," in IEEE Sensors Journal, vol. 15, no. 6, pp. 3281-3292, June 2015, doi: 10.1109/JSEN.2014.2388153.

[2] S. Sukkarieh, Low cost, high integrity, aided inertial navigation systems for autonomous land vehicles, PhD Thesis, Australian Center for Fields Robotics, University of Sydney, 2000.

[3] Francois Caron, Emmanuel Duflos, Denis Pomorski, Philippe Vanheeghe, GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects, Information Fusion, Volume 7, Issue 2, 2006.

[4] Wang, S.; Deng, Z.; Yin, G. An Accurate GPS-IMU/DR Data Fusion Method for Driverless Car Based on a Set of Predictive Models and Grid Constraints. Sensors 2016, 16, 280.

[5] Chao, H., Gu, Y. & Napolitano, M. A Survey of Optical Flow Techniques for Robotics Navigation Applications. J Intell Robot Syst 73, 361–372 (2014). https://doi.org/10.1007/s10846-013-9923-6

[6] K. McGuire, G. de Croon, C. de Wagter, B. Remes, K. Tuyls and H. Kappen, "Local histogram matching for efficient optical flow computation applied to velocity estimation on pocket drones," 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 3255-3260, doi: 10.1109/ICRA.2016.7487496.

UNIVERSITY OF MICHIGAN

Questions?