Functional Requirements Document

GPS + Inertial Sensor Fusion (GISF)

Date:
November 3rd, 2013

Team Members:
Aleksey Lykov,
William Tarpley,
Anton Volkov

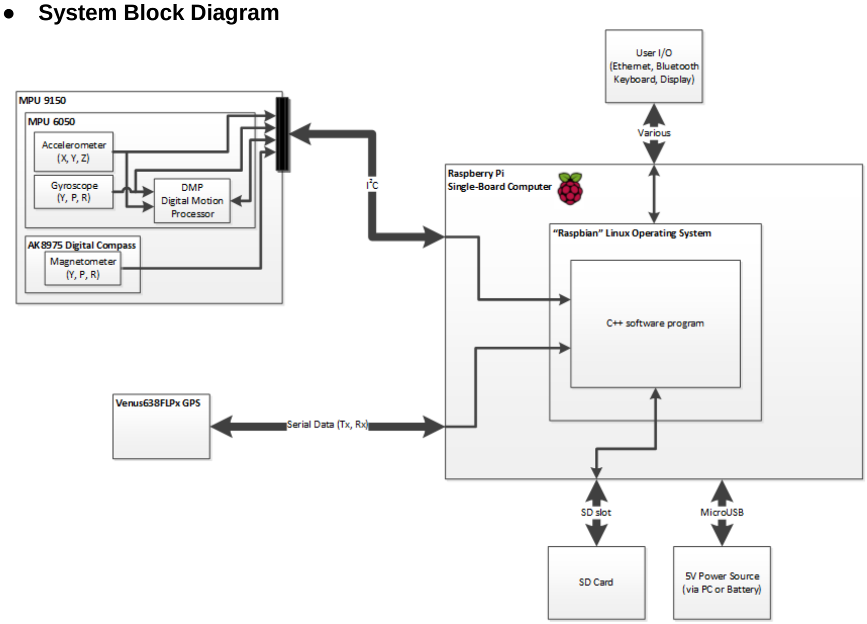
Advisors:
In Soo Ahn,
Yufeng Lu

- **Introduction**

  An inertial navigation system (INS) or inertial measurement unit (IMU) is a form of dead reckoning navigation system that uses a combination of accelerometer and gyroscope sensors working in concert to detect displacement relative to a starting point. The system measures both linear accelerations given by its three-axis accelerometer and angular velocity changes from the gyroscope measurements. World referenced-frame acceleration data can then be integrated to calculate the velocity and position of the sensors over time, but because the INS can only measure motion in relation to a starting location, the initial position must be supplied by some outside system (in the case of this project, a Global Position Sensor). Additionally, to compensate for the drift in the inertial navigation system caused by various defects in inertial sensors, this outside reference (GPS) must be polled occasionally to correct for the position error.

  Using a Raspberry Pi microcomputer as the base system and an MPU 9150 IMU, an inertial navigation system will be developed. Kalman Filtering and GPS will be used to complete a “strapdown solution” - a closed-loop system which can self-correct for error.

- **System Block Diagram**



- **Functional Description**

  The main sensor used for the project is the MPU 9150 Inertial Measurement Unit (IMU). This chip sends out nine axes of data: x-acceleration, y-acceleration, z-acceleration, yaw, pitch, roll, and three axes dedicated to magnetometer data. The three-dimensional acceleration and gyroscope data is used for data acquisition. Additionally, the chip has the capability to output the quaternion as well as real-world adjusted data which combines the data from the gyroscope and accelerometer to get real-time accelerations for yaw, pitch and roll. If needed, the magnetometer can be used to help calibrate the gyroscope to compensate for some of the drift.

  A GPS unit will be used to give the program its initial position and to correct for error through the operation of a Kalman Filter.

  The heart of the system will be the Raspberry Pi microcomputer. It runs C or C++ code for data acquisition and data processing. The IMU and the GPS are semi-permanently attached to the I/O pins of the Pi, which read in data from both devices after which the data is time stamped and saved to SD card. Eventually, the Pi will also perform real time filtering of the data instead of just data acquisition in order to calculate current system position.

- **Functional Requirements for System Operation**

  - System shall be ready to start navigation 30 seconds after being started
    - 30 seconds to acquire GPS satellite position lock while outdoors
    - 20 seconds to simultaneously stabilize gyroscope
  - System shall rely primarily on inertial measurements, using GPS external positioning input only to correct for drift [1]
  - After one minute of operation without GPS data, the system shall report a position within either 5% or 1 meter of the true displacement, whichever is greater
    - If the user travels 100 meters in one minute, the system shall report between 95 and 105 meters total displacement
    - If the user travels 0 meters in one minute, the system shall report a distance “travelled” – a drift – of less than 1 meter
  - While maintaining a GPS signal, the system will report a position with a degree of accuracy with less than 5% error (if 100 kilometers are travelled in one hour, the system will report between 95 kilometers and 105 kilometers travelled)
  - If the system loses a steady GPS signal, it shall alert the user via the display interface
  - The system shall initially save all acceleration and GPS data to text file, but the final product will be able to integrate into velocity and position data (and also correct for errors in real time)

Commented [1]: Anton Volkov:
https://en.wikipedia.org/wiki/Extended_Kalman_filter for later reference.

Commented [2]: Anton Volkov:
Maybe rephrase this... if GPS, ±5%, if no GPS, ±X%

Anton Volkov:
idk.

wtarpley:
I don't mind the wording but there should be multiple points... So after 1 minute, 5minutes, and 60minutes (we're fucked)

Commented [3]: Anton Volkov:
This seems like a high expectation.

Anton Volkov:
But I think we can work towards it.

wtarpley:
I'm fine with this but if we don't meet this goal it will likely reflect in our grade.

Anton Volkov:
That's a good point. So it sounds like we'll be meeting it regardless.

alykov:
5% of what? "Position" is not a unit of measurement. Can we just say "within 5 meters"?

Anton Volkov:
"5% of the actual position"

Anton Volkov:
if you travel 100km, 5m is .005%.

Anton Volkov:
To elaborate, I mean within 5% of the actual displacement in any unit of measurement. The system will have more error with more time.

alykov:
Yeah, but that is silly. What if the user took one step. If he traveled one foot and we did not measure it, then we are off by 12 in. 5% of that is a small number. I do not like such scaled measurements. What if the user is moving in a car at ten meters per second? After 60 seconds he will have traveled 6000 meters. 5% of that is a whopping 300 metrs. Are you OK with that level of error?!

...

Commented [4]: Anton Volkov:
1 meter ok? We can make this .5 or .1 meters for stationary...

Anton Volkov:
I think 1 meter is plenty ok.

- The system shall re-acquire a GPS signal if it is lost within 30 seconds of entering an unobstructed outdoor environment [4]
- The system shall operate continuously (without any crashes or lapses in data acquisition) on battery for a minimum of 2 hours
- Maximum acceleration: ±2g, ±4g, ±8g, or ±16g selectable by user [2]
- Maximum angular rate: ±250 degrees/second at max resolution [2]
- During operation, the system shall display the following information to the user via LCD:
  - Whether a GPS lock exists and the last known GPS coordinates
  - The displacement from start [m, X/Y/Z]
  - The "odometer" distance from start [m]
  - The current velocity [m/s]
  - The current acceleration [m/s^2]
- Additionally, the system shall record the following information to text file on SD during operation:
  - Early on:
    - Each timestamped acceleration reading
    - Each timestamped GPS coordinate
  - Later, will also include:
    - Each timestamped velocity and position reading
    - Kalman-filter-adjusted acceleration readings
  - Finally, will also include:
    - The system's best timestamped estimates for X, Y, and Z acceleration, velocity, and position from start
- The system will maintain a post-processed position update rate of at least 5 Hz
  - Acceleration measurements shall be taken at a minimum of 100 Hz [3]
  - GPS measurements shall be taken at a minimum of 1 Hz when GPS signal is present [4]

Commented [5]: alykov:
We are vouching for the system's power consumption here. Can we confirm that the pi+sensors can run for two hours on a battery pack that we can acquire?

Anton Volkov:
Right. The battery pack I currently have (the white one, USB output) is 12Ah, which should theoretically supply enough power for 6 or more. I don't think we've ever tested for stability, though. And long-term (>2 hour?) testing would be good start).

alykov:
Alright. Let's begin tests by dropping it in the toilet and then go from there

Anton Volkov:
Haha we'll do than and catch it just as it enters the peoria river.

Commented [6]: Anton Volkov:
Just looked this up on sparkfun (https://www.sparkfun.com/products/11486) it would be good to confirm that we're running at this rate.

alykov:
Did you look that up before or after you wrote it? Where does that number come from? Is this a generally accepted rate?

Anton Volkov:
It's in the specs (between 250dps and 2000dps selectable. I think we'd want the highest resolution (250dps) for mroe accuracy)

wtarpley:
we could also offer this is an option... although I'm trying to think of a reason why

Anton Volkov: ...

Commented [7]: Anton Volkov:
Any other big things? I think accuracy is our main focus
How about functionality? Did we outline specifically WHAT we want it to measure? As far as requirements go, that is.
Will it be
3d
spaaaaaaace?
ok. I think we should outline what it displays during a test, but we do still probably want a log of accel/Vel/position in case we want to plot it on a map afterwards
Since this is a sensor, it should have a specific output based on the inputs. Somewhere, we need to explain whether the output is a coordinate in XYZ, GPS or...cartesian?Polar?Rectangular?
Right. The output should be usable by us… I think that if we really just care to plot on Google Maps, we can make the system compute an approximate GPS coordinate…. maybe not. idk. ...

- **References**

[1] Rockwell Collins. GuS Integrated Sensor Suite [Brochure]. 2012 Warrenton, VA. Internet: http://www.rockwellcollins.com/~/media/Files/Unsecure/Products/Product%20Brochures/Controls/Flight%20Controls/Athena%20GuS/Athena%20GuS%20data%20sheet.aspx, 2012 [11/03/2013].

[2] Sparkfun. 9 Degrees of Freedom - MPU-9150 Breakout. Internet: https://www.sparkfun.com/products/11486, 05/14/2012 [11/03/2013].

[3] Invensense. MPU-9150 Product Specification. Internet: http://invensense.com/mems/gyro/documents/PS-MPU-9150A-00v4_3.pdf, 09/18/2013 [11/03/2013].

[4] Sparkfun. Venus GPS with SMA Connector. Internet: https://www.sparkfun.com/products/11058, 01/25/2011 [11/03/2013].