|  |  |
| --- | --- |
| Project Name: | MailBird: An Autonomous Delivery System |
| Team #, Members: | Team 1, Ben Smith, Hugh Dillon, Hunter Thorington, Rick Holloway, Zac Hawkins |
| Report Date: | February 19, 2014 |
| Project Description: | A landing system that can guide a vehicle using IR LEDs within 1 inch of a target. |
| Cycle (1, or 2): | Cycle 1 |
| Cycle Intent: | Build a working prototype of IR module and use to mimic loiter behavior over LED station |

**TASKS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Planned |  |  | Actual |  |
| Task # | Task Description (Add rows as needed) | Cycle planned for completion | Total planned hours | Planned hours this cycle | Status (% complete) | Actual hours this cycle | Total hours |
|
|
| 1 | Team management | 2 | 60 | 30 | 26.67% | 8 | 8 |
| 2 | IR land control method | 1 | 120 | 120 | 47.08% | 56.5 | 56.5 |
| 3 | IR camera implementation | 1 | 40 | 40 | 162.50% | 65 | 65 |
| 4 | Ground Station control method | 1 | 40 | 40 | 65.00% | 26 | 26 |
| 5 | Landing station | 2 | 20 | 10 | 10.00% | 1 | 1 |
| 6 | Reports | 2 | 180 | 80 | 13.75% | 11 | 11 |
| 7 | Marketing display | 2 | 40 | 0 | 0.00% | 0 | 0 |
| 8 | Integration of components | 1 | 100 | 100 | 107.00% | 107 | 107 |
|  |  | **Planned Total** | 600 | 420 | **Actual Total** | 274.5 | 274.5 |

1Planned Total should equal (# of team members) x (10 hrs. per week) x (Cycle 1 weeks 6) + Cycle 2 weeks (6) = 12 weeks).

2Assumes 5 hours per week for 12 weeks. Should be mainly team leader(s).

**TEAM MEMBER HOURS**

**Record # of hours each person spent on each task this week, then total by week, cycle, and project.**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **Task3** |  |  |  |  |  | **Total Hours** |  |
| **Name** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **Week** | **Cycle** | **Project** |
| **Dillon, Hugh** | **2** | **--** | **5** | **--** | **--** | **--** | **--** | **15** | **22** | **64.5** | **64.5** |
| **Hawkins, Zac** | **--** | **--** | **4** | **--** | **--** | **--** | **--** | **5.5** | **9.5** | **41** | **41** |
| **Holloway, Rick** | **--** | **--** | **13** | **--** | **--** | **1.5** | **--** | **15** | **29.5** | **73.5** | **73.5** |
| **Smith, Ben** | **--** | **--** | **--** | **--** | **--** | **1** | **--** | **7** | **8** | **40.5** | **40.5** |
| **Thorington, Hunter** | **--** | **--** | **8** | **--** | **--** | **--** | **--** | **2** | **10** | **55** | **55** |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **TOTALS** | **2** | **0** | **30** | **0** | **0** | **2.5** | **0** | **44.5** | **79** | **274.5** | **274.5** |

**Accomplishments since last status report:**

* The IR camera sensor is now completely integrated into the ArduPilot software. We can see data running on the oscilloscope and can read that data into our control system elsewhere in the software.
* A test environment to debug the I2C connection was created and used to determine the cause of communication errors between our IR camera sensor and the I2C driver in the ArduPilot code. The issue seems to be that sometime between the time we initially enable interrupts and the time we attempt to read data from the camera, the interrupt enable flag was being unset. We solved this by simply setting it again before reading, but

**Obstacles encountered since last status report and actions to deal with same:**

* The quadcopter does not seem to want to arm using the RC transmitter now that we have potential to fly it using custom code. This seems to imply that we have something incorrect in our software still, as reverting back to old code will allow the RC transmitter to arm. We have to investigate why this is the case this week.
* Once the entirety of the ArduCopter code is running, serial print statements become very CPU expensive and it is difficult to debug timing and control with all the excess load serial printing causes. We’re going to need to get creative to verify certain aspects of our code. While MAVLink requests may be still be made via USB (as far as we know), it still may be difficult to return variable values and debug timing, which is crucial for flight code.

**Risks facing the project and actions to deal with same:**

* As we said last week, we’re doing a significant code addition. We’re using the best software tools we know to maintain the modified Arducopter codebase. Code checkout and change lists are a good way to mitigate the risk of a lot of wasted time editing code and fixing code that used to work.
* After modifying the flight code, there may be potential for unbounded behavior from the quadcopter while it’s in the air. A tether of some kind is needed to ensure that it doesn’t fly away from us. This is a worse scenario than crashing because if it leaves we cannot track it in time to get it back.
* Designing good tests that actually prove something is difficult. We’re writing down exactly what data we’re taking and how we plan to prove functionality as we add these tests.

**Objectives for the next week:**

* Develop a test setup to determine why our ArduCopter will not arm.
* Develop a good way to see what’s going on inside the ArduCopter to verify that we are indeed controlling the helicopter correctly. This will be needed to show that we have added an effective code addition.
* Begin working on documents required at the conclusion of Cycle 1, which is in 2 weeks.