5/10/16

MOTIVATION

Safety is what defines the aerospace industry. (1 accidents / 4 million flights)

UAVs are part of the aerospace industry, but still in the development phase

Taking the pilot off the vehicle ~~creates~~ generates some technical challenges (comms, inertial cues, flight rules...) that restrict the usage of UAVs to their full potential.

Regulations limit the operation due to those problems

GOAL: Improve safety on-flight with an Obstacle Collision Avoidance System

↳ Problem statement
- Improve active safety during flight
- Applicable to existing UAVs
- Only modify behaviour when there is a safety issue

METHODOLOGY: SE for a systematic analysis of the problem

OBJECTIVES: Following SE guidelines → Design, Implementation and Testing

RESULT: OCAS that is reliable, modular (expanded to any specific need) and open (for further research)

DESIGN
- What must be done? → Requirements
- What are the elements needed? → Logical decomposition

REQUIREMENTS: A statement or characteristic of something that is needed
Used to define ~~the~~ the details of the problem

Skip
- Come from: Customer needs → Motivation
  - Regulations
  - Goals and objectives
  - Assumptions and constraints

The OCAS shall:
- Improve safety
- Avoid collisions) derived
- Work independently of UAV (modular design)
- Not interfere with UAV functions (just a safety layer)
- Operable after short training (potential costumers' interest)

LOGICAL DECOMPOSITION: Several tools. Final goal is to identify the building blocks of the system and the interfaces between them.

FFBD: Sequential order of the functions to be accomplished during the mission
"Detect" and "avoid" are the most critical functions
Detect: Endless loop until situation is considered unsafe → Req. of not interfering

PBS: Identifies the elements and components needed to perform the defined functions
Most important elements: Sensors, computer and software

N^2: Defines the interfaces between the elements on the PBS
Output information from any block goes ~~at~~ its horizontal row, while inputs go ~~at~~ the vertical column

Without OCAS: Green loop: Pilot → UAV → Pilot
With OCAS: Classic control problem: CPU → ... → Sensors → ... → UAV

IMPLEMENTATION: How will the system meet the requirements?

1st: define the system architecture with the OCAS integrated (onboard the UAV)
Before moving on, the specific components need to be chosen

SENSORS: Trade-off study, to choose the most appropriate alternative for the project
Sonar comes up as best because light weight, low cost, simple processing
However, cons: Noise, ghost signals, multipath errors

COMPUTER: RPi → Full computer, small footprint. Very FLEXIBLE (software and interfaces)

Skip
OTHER COMPONENTS: Battery → According to independency requirement
Wifi → Wireless comms. with GCS
Platform → DJI F450 weighting approx. 1kg

HARDWARE INTERF.: RPi is the brain of the system
USB to UAV. USB to Wifi to GCS
GPIO to sonar. GPIO can be directly controlled via software, setting each pin to HIGH or LOW
MAVlink → Communication protocol with UAV. Allows reading its state and sending navigation commands
SSH → Remote ~~cont~~ operation of the RPi via Shell

SOFTWARE INTERF.: Schematic derived from the N^2 diagram
1st layer: Raspbian OS → Utilities for interfacing with peripherals (WiFi, USB, GPIO) + Ability to run other programs
2nd layer: Python environment → Interpreter to run applications programmed in Python
GUI: Easy execution of MAVproxy and the Control Script
MAVproxy: Redirects information to and from the UAV (MAVlink)
Script: Implements the missing elements from PBS
→ Explain

SCRIPT: Consists of several files. Connected by the main.py
Flowchart follows ~~the~~ sequence defined in the FFBD

Skip
Evaluate distance: Signal sequence according to technical documentation of the sonar
Measured distance depends on ambient temperature. Negligible

Compute velocity:
Distance measurements are often noisy. Differentiating (high-pass filter) a noisy signal dramatically increases the errors when propagating into the future
Taking a bigger stencil dampens the disturbances, while maintaining responsiveness

Level of threat:
Knowing current speed and velocity w.r.t obstacle, predict time to collision
There is risk of collision when time to stop the UAV is bigger than t to collision
Hence, avoidance manoeuvre should be triggered when $t_{safe} \leq 0$

Avoid obstacle:
Now → Stop the UAV (loiter). Simple go_to commands (relying on GPS)
Future → Real time control algorithms. Take data from sonars + other sensors

TESTING: Does the proposed solution work?
Testing phase is critical to ensure that the system meets the specifications
Every element should be tested. Here only focus on the final UAS test
Platform: F450 modified to fit additional components of the OCAS
Setup: Avoidance manoeuvre relies on good GPS signal.
Procedure: Following FFBD. Avoidance manoeuvre is to climb 4 m, because very visual
Results: Explain changes of ~~mo~~ flight mode (if there is time)

CONCLUSION

To conclude: Has the goal of increasing the safety during flight been met?
I belive that, while there is a big potential for improvement ~~in~~ the implementation phase, the proposed solution serves as a proof-of-concept to demonstrate that the conceptual design represents a robust baseline on which to base future work

Finally, I would like to mention that Obstacle Avoidance is just ~~a possibi~~ one of the alternatives available to improve UAV safety. Ultimately, the aim is to reach aerospace industry standards also for Unmanned Aircraft, so that they can be used to their whole potential, which is not small at all.