Designing for Human–Automation Interaction: Abstraction–Sophistication Analysis for UAV Control

Matthijs H.J. Amelink [*], Max Mulder and M.M. van Paassen [†]

Abstract—**The abstraction–sophistication analysis has been developed in extension of the abstraction hierarchy to aid the design for effective human–automation interaction for vehicle control systems. The new analysis framework is applied to the mini UAV system being developed at the D–CIS lab and TUDelft.**

Keywords: ***cognitive systems engineering, abstraction hierarchy, functional analysis, UAV system, abstraction–sophistication analysis***

1 Introduction

In 2005 the D–CIS lab and the department of Aerospace engineering at TUDelft started a collaborative project to design and build a mini Unmanned Aerial Vehicle (UAV) system. The aim of the project is to deliver a research UAV system capable of controlling multiple UAVs simultaneously. The UAVs will be used as test beds for the development of advanced UAV controls e.g., high level autonomy in complex missions and swarming principles. Currently the software on the Ground Control Station (GCS) supports mission planning for and control over multiple UAVs.

A high level of autonomy is desired to relieve the operator from workload per UAV and thereby allow him/her to operate multiple UAVs simultaneously. Modern automation and computer systems provide designers with virtually unlimited degrees of freedom to design automation and interfaces for operators. Despite the benefits, automation has the potential to adversely increase complexity and induce human errors. Guidelines and design paradigms are needed to limit *automation–induced complexity.*

Cognitive Systems Engineering (CSE) and Ecological Interface Design (EID) provide guidelines for design of systems with effective human–machine coupling[4][6]. These guidelines have been applied in high risk domains: nuclear power plants, aviation and medicine[3][2]. Typically, CSE has been used to analyze *existing* systems with the goal to improve them. In contrast, the UAV system is designed from the ground up guided by insights gathered through experience with CSE and EID.

It is hypothesized that for effective human–automation interaction, humans and the automation should base their reasoning and control actions on the same domain representation[1]. To derive this model for automation an ecological approach is taken. The term *ecological* denotes the relation between an organism and its environment and is used to emphasize the user (or operator) in relation to the goals (s)he wants to achieve in the environment. The ecological approach suggests that work analysis should begin with, and be driven by, an explicit analysis of the constraints that the environment imposes on action [5].

The abstraction hierarchy is fundamental to this approach. It is used to "produce a generalized representation of the 'work domain' in terms of its inventory of objectives, functions, activities and resources – all of which constitute the element of the landscape in which the staff operates."[4]. The abstraction hierarchy maps out the means–ends structure in a work domain, it is used to identify functions and their structure useful to human operators in achieving their goals.

This paper describes how the abstraction hierarchy is extended to analyze a to–be–built UAV system and introduces the Abstraction–Sophistication Analysis (ASA). It is explored how the analysis can be used to design the automation (in addition to the interface) to limit automation induced complexity. The UAV system is used to exemplify, due to space constraints the analysis cannot be shown in full detail.

2 UAV System Description

Our mini UAVs are typically operated a few kilometers from the GCS on an operational/ad hoc basis. The fixed wing UAVs are model airplanes that have a wingspan up to 260 cm, a mass up to 2 kg, and are electrically powered. A planned rotary wing UAV has a rotor diameter of around 200 cm, a maximum takeoff weight of 7 kg, and

[*]Thales Research & Technology NL, D–CIS lab. Delftechpark 24, 2628XH Delft, the Netherlands.
[†]Delft University of Technology, Faculty of Aerospace Engineering, Kluyverweg 1, 2629HS Delft, the Netherlands

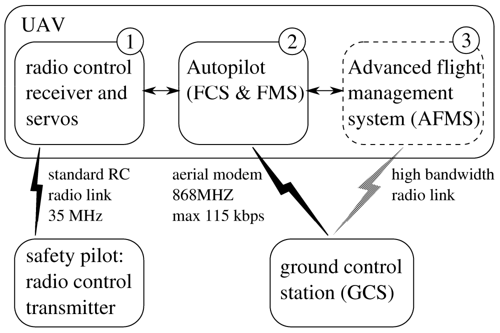
UAV
1
radio control receiver and servos
2
Autopilot (FCS & FMS)
3
Advanced flight management system (AFMS)
standard RC radio link 35 MHz
aerial modem 868MHZ max 115 kbps
high bandwidth radio link
safety pilot: radio control transmitter
ground control station (GCS)

Figure 1: Three system group definition.

a maximum payload of 2 kg. The ground control station is based on a laptop computer. Apart from the autopilot board, the UAV system is based on commercial off the shelve components.

Safety is a main design requirement. The inner flight control loops have to be highly reliable and independent of potentially unpredictable experimental software and hardware. Therefore three system groups were defined shown in Figure 1. The first group contains the standard radio–control equipment that allows the UAV to be flown like a regular model aircraft. System group 2 is the custom designed autopilot that runs the Flight Control System (FCS) and the Flight Management System (FMS). The autopilot hardware and software are of a non–experimental nature and form a high reliability system group. The FMS has basic navigation functionality, it is capable of waypoint navigation and has a 'return home' function in case communication fails. The third group, the Advanced Flight Management System (AFMS) is of a more experimental nature. It is a separate computer board with the needed processing power to run collision avoidance algorithms and process camera images for on–board vision. If the AFMS fails the FMS in system group 2 will take over. A safety pilot is standing by to take control when system groups 2 fails.

Hardware choices were fixed based on budgets and availability, and are regarded as external constraints for the analysis for automation design. Figure 2 shows the basic hardware architecture based on the three system groups. The autopilot has one processor running both the FMS and FCS software and has the flight sensors for flight control. Among the flight sensors are: the 6 degrees of freedom Inertial Measurement Unit (IMU), the 4Hz GPS receiver, magnetometers, a barometric altimeter and will include a pitot tube for airspeed measurement in the near future. The GCS and the autopilot communicate over a long–range modem with up to 15km range. This link is used to upload onboard control gains in–flight (FCS) and to upload waypoints to the FMS. UAV state information is streamed to the GCS. A shorter range but high bandwidth link will be used to communicate with the AFMS. The UAV software package is also developed

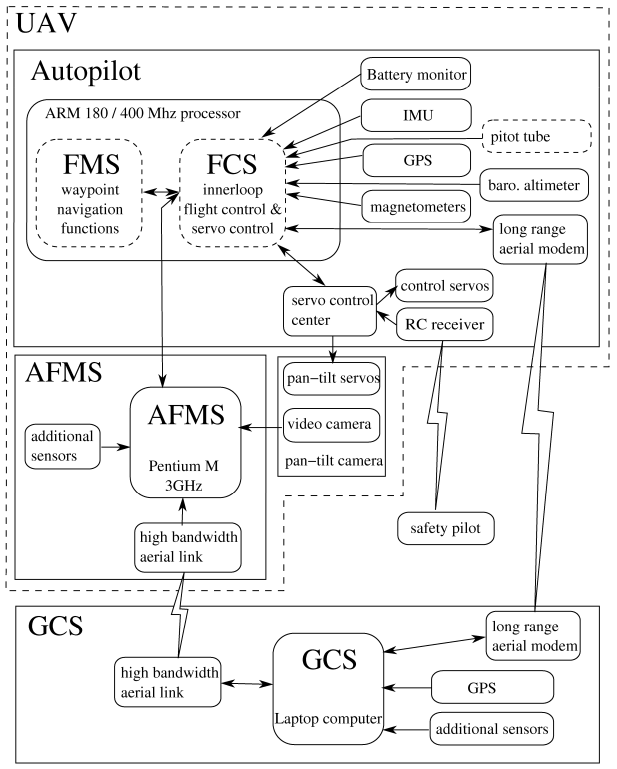

Figure 2: Basic hardware architecture.

in–house. The current version supports three main configurations: real flight, hardware–in–the–loop simulation and fully simulated. The latter mode allows rapid development of automation and control concepts. Software design choices e.g., choice of algorithms, and interface design are supported by the analysis. The next section discusses the structure of the abstraction–sophistication analysis.

3 Abstraction–Sophistication Analysis

3.1 Principles

Six levels of control sophistication are introduced to capture the control layers that are typically found in vehicle control like in our UAV system. These levels are characterized by the following:

- Each level of sophistication is a layer of control based on the levels below, achieving more sophisticated control of the system.
- Each level of sophistication holds functions and concepts for control specific to that level.
- Lower levels describe the inner control loops and higher levels describe the outer control loops of the total system.
- At each level of sophistication five levels of abstraction are used to find the means–ends structure of the control problem described at that level of sophistication. These levels correspond to the levels

Rasmussen found for process control.

For example, in Figure 4 an overview of the ASA for the UAV system is shown. The concepts on the 'mission level' form a description about the mission objectives and how those can be achieved. 'Navigation' is a necessary concept at the 'mission' level needed to achieve missions. 'Navigation' is described at the level below in terms of the concepts needed to provide navigation capability.

Rasmussen and Vicente use structural decomposition (part–whole relation) as a second dimension to the abstraction hierarchy for the process control domain [4][5]. The resulting abstraction–decomposition space is a two–dimensional matrix where every cell describes the whole system at a different level of resolution(part–whole) and meaning(abstraction). In contrast the ASA produces a two dimensional matrix where the description at each level of sophistication encompasses all the levels below that level but not the whole system. Structural decomposition also plays a part in abstraction–sophistication analysis and is depicted by splitting the ASA into multiple diagrams. As Figure 5 shows, payload control can largely be analyzed in a separate ASA. Control over UAV and payload are merged on the 'mission' level where, as seen in Figure 4 two concepts are inserted: 'navigating UAV' and 'controllable camera'.

The levels of the abstraction hierarchy are applied in the same way as was originally done for process control. The following guidelines apply. At the 'functional purpose' level the purpose of that level of sophistication to its environment is given. The environment is the above level of sophistication. At the 'abstract function' level, *priority measures* for allocating resources are represented. With automation design in mind this representation is defined in terms of a transformation with defined inputs and outputs. This can be human information processing and/or a computer algorithm where the latter has to be well defined as will be shown in section 3.3. At the 'generalized function' level, functions of the system are identified without physical implementation. At the 'physical function' level, functions of the system are identified according to their underlying physical processes. At the 'physical form' level, the physical properties and layout of system components are specified. Physical interactions can play an important role, for example placing the GPS antenna on the belly of the aircraft will degrade satellite signal reception and will degrade overall UAV performance.

The 'physical form' level is not divided by the levels of control sophistication because it is the most concrete level that represents how the system is physically implemented. Levels of sophistication become useful when abstracting away from the physical form. For example at the 'physical form' level a building has no more meaning than a pile of bricks but on higher levels of abstraction that pile of bricks gets meaning to the UAV system. At the 'aviate' level of sophistication a building means an obstacle that has to be avoided and at the 'mission' level of sophistication that same building could be a target for observation.

The ASA assists in mapping out the needed control structure and interface in terms of functions for UAV and payload control. The following section explains how the ASA is applied to the mini UAV system.

3.2 UAV system analysis

The six levels of sophistication identified for the mini UAV system are: the flight level, the flight control level, the aviate level, the navigation level, the mission level and the joint mission level. The levels are discussed below.

At the 'flight' level the functions and concepts needed to achieve a flying platform are described, it is the basis for the whole system. The functional purpose of this level of sophistication is to provide a 'flying platform' as a concept at the next level of sophistication. The transformation that describes how this is achieved is found at the 'abstract function level'. The aircraft state development over time is denoted here. The generalized function level describes the most familiar concepts of flight: lift, propulsion, stability and controllability. How these functions are physically achieved is described at the 'physical function' level. The 'physical form' level describes the physical properties of the components that achieve the physical functions. For example the wings (what) can generate lift (why relation) because they have certain dimensions: wingspan and airfoil (how relation).

The functional purpose of the 'flight control' level is to provide a controllable UAV platform through position vector control. The functional purpose of the 'flight level' has become a generalized function, the flying platform is considered a single concept at the generalized function level. Other generalized functions are those performed by the FCS: altitude hold, heading hold, state estimation, etc. The generalized functions achieve the transformation described at the 'abstract function' level where the estimated state and desired state are inputs to the control action computation. When implemented robustly this transformation achieves the functional purpose of the 'flight control' level: a UAV that is controllable with position vector control commands. At the physical function level we find the flight sensors (GPS, IMU, altimeter, etc.) and the autopilot. At the 'physical form' level the physical properties of the autopilot and sensors are represented.

Once a controllable UAV platform has been established, the next level of sophistication deals with where to fly in direct relationship to the environment: aviating. The 'aviate' level of sophistication describes the concepts concerned with piloting or aviating and provides an aviating UAV that will fly in preferred areas, stay out of no–fly zones and avoid hazardous areas. The 'aviate' automation is executed by the AFMS. Going from 'physical form' to 'physical function': objects in the environment are represented as trees, buildings, masts, etc. At the 'generalized function' level these objects are obstacles need to be avoided because they form a hazard to the UAV or because regulation will not allow flight near a royal palace. Together, the obstacles set constraints on possible flight paths and a safe and efficient path has to be computed either by automation or the human operator. The computational transformation is represented on the 'abstract function' level. The 'aviate' level is further exemplified in the next section.

At the 'navigate' level of sophistication the UAV system's navigation concepts are found. Providing a 'navigating UAV' is the functional purpose. This is achieved by the abstract navigation logic executed by the FMS or the AFMS. The 'generalized function' level describes the concepts taken into account by the navigation logic such as: no fly zones, preferred routes and airspace regulations. The 'physical function' level describes the physical functions such as an airport that is represented as a no–fly zone at the generalized function level.

The 'mission' level contains the concepts associated with the mission. The navigating UAV is inserted as a single concept on the generalized function level. The functional purpose is achieving the mission objective in a safe and efficient manner. At the abstract function level the mission logic is described. This includes how the UAV is to be flown during the mission in coordination with control over payload. Considering a search mission, the mission logic would be to first compose a flight plan that brings the UAV to the search zone. At the search zone the UAV will fly the appropriate search pattern that gives the right coverage with the camera. The pan–tilt camera is controlled to cover the whole area. Payload control (see Figure 5) is merged with UAV control at this level. Note that the properties of the payload chosen in design phase, e.g., a large and heavy camera, will impact the UAV control at the 'physical form' level of abstraction and 'flight' level of sophistication.

The 'joint mission' level describes concepts that apply to the coordination of multiple missions with multiple UAVs and is not further detailed here.

A similar ASA can be made for payload control. Two levels are identified below the 'mission' level labeled 'sensor' and 'sensor control' as shown in Figure 5. The same relations across the levels hold as discussed above for the UAV analysis. At the 'mission' level the two ASA diagrams are connected.

3.3 Implementation of the Aviate Level

This section discusses how the analysis on the 'aviate' level helped define the automation and the interface to the operator. The current implementation of this level achieves autonomous avoidance of known obstacles stored in a database during simulated flight while the operator is supported to see, predict and understand the automation logic in the interface.

The UAV can be instructed to fly to a certain point that can be a waypoint in the flight plan, an instruction from the mission logic, or a request from the operator. The UAV then needs a safe and efficient path to that point. For a human operator this is typically a task with high cognitive workload and needs to be automated. However, if the automated path planning is unpredictable, hard to understand or seemingly inefficient, the automation will in fact increase workload. The aim is to design automation and interface based on a domain representation shared with the operator based on the structure of the work domain. This is where the abstraction hierarchy comes in.

The approach chosen is based on the A* pathfinding algorithm. A* finds a path from the present location to the next waypoint based on a virtual cost map. The cost map is an abstract representation of obstacles, no–fly zones, areas to avoid and preferred flight zones. The cost variable is an abstract measure used to find the best path, which is the path with the lowest cost associated with it. Buildings and other hard obstacles have an infinite cost with the effect that a path cannot be planned through it. To limit the needed computational resources the resolution of the cost map in vertical direction is limited to two layers. The bottom layer has a height set to 50m so the top layer will clear most obstacles. In both layers hard obstacles like buildings, trees and masts exist, but also soft obstacles like a residential area that has some cost associated with it. The UAV is allowed to fly over those areas but it is not preferred, while preferred UAV routes have no cost. The length of a path also has costs associated with it to give preference to short paths. The A* algorithms weights all the costs and finds the cheapest path. If a residential area is small, the algorithm is likely to plan a path around it but if it is large, the algorithm is likely to find a short passage over this area.

The A* algorithm is straight forward and not complex in its behavior, thus easy to understand. The cost map shapes the computed path and thereby the UAV's behavior. These constraints are visualized in the interface and the operator can easily see, predict and understand the algorithm's solution based on the constraints. Figure 3 shows the map view presented to the operator when viewing the 'aviate' level of the control interface. In addition to obstacles, virtual shapes can be added to the cost map. These shapes will further constrain the behav-

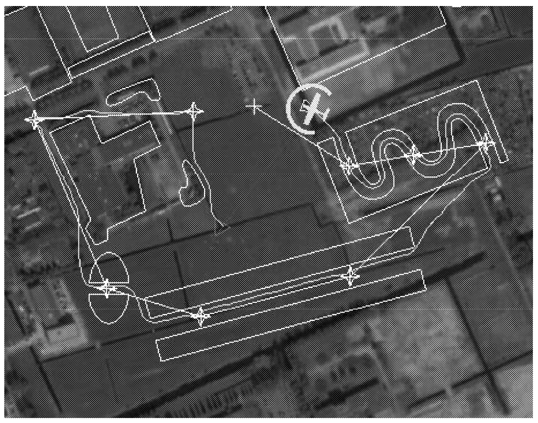

Figure 3: Map view of the GCS interface. The outline shapes correspond to obstacles (buildings and trees) and to virtual constraints that can be used to shape the flight path.

ior of the UAV, which can be useful for a specific mission or adding information that is not directly supported in the interface. For example, in the case of a forest fire, the airspace directly above and downwind of the fire can be blocked with a virtual constraint to prevent the UAV from flying into hot and hazardous air. Figure 3 shows three virtual constraints shaping the flight path.

4 Discussion and Future Work

The ASA is a tool to map out functions, concepts and constraints in system. Our UAV system is designed to allow the operator to interact at different levels of sophistication and different levels of abstraction. The interface allows interaction with many of the concepts and functions discovered with this analysis. The automation is being designed to work with the same functions and concepts creating transparent automation. What automation does is visible to operators and vice versa.

Figures 4 and 5 are simplified due to available space. The fully detailed analysis yields many concepts that can severely clutter the interface. We find it useful to structure the interface according to the levels of sophistication and show only the levels at which the operator wishes to interact. For example, if an operator sees one UAV go astray, (s)he could view the 'aviating' level interface and see how the A* algorithm found a path on the cost map. If the solution does not make sense, something could be wrong and (s)he could troubleshoot the lower levels of abstraction and possibly find a problem with the AFMS.

The ASA allows keeping track of how constraints propagate through the system. We have experienced this to be valuable during the design and engineering phase. To illustrate: a helicopter UAV seems an attractive option because its control is conceptually simpler at the 'navigate' and 'mission' levels. A helicopter can hover and observe from a stationary position, and do vertical takeoffs and landings. However, the complexity grows on the 'flight control' level where platform stabilization and reliability are defined. Also the inherent safety aspects propagate upwards: a 7 kg helicopter storing additional kinetic energy in its rotor blades is inherently more dangerous than a 3 kg styrofoam plane carrying the same payload. In our application these constraints outweigh the apparent simplicity of helicopter control.

The basic system architecture has been implemented. The 'flight', 'aviate', 'navigate' and 'mission' levels of sophistication are explicitly represented in the interface. The mini/micro UAV community has very positively reacted to these concepts at the Third US–European Competition and Workshop on Micro Air Vehicles. Up until now real flights have been carried out with FCS and FMS providing only basic navigation functionality. Future work involves implementing the AFMS hardware onboard for in–flight collision avoidance.

Acknowledgements

The research reported here is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant nr: BSIK03024.

References

[1] Matthijs H.J. Amelink, Max Mulder, M.M. van Paassen, Gavan Lintern, and Iya Solodilova-Whiteley. Models for automation: Learning from autopilot design. In *Proceedings of the International Conference on Human – Computer Interaction (HCI-Aero)*, 2006.

[2] Matthijs H.J. Amelink, Max Mulder, M.M. (René) van Paassen, and John M. Flach. Theoretical foundations for a total energy-based perspective flight path display. *The International Journal of Aviation Psychology*, 15(3):205–231, 2004.

[3] Nick Dinadis and Kim J. Vicente. Designing functional visualizations for aircraft systems status displays. *The International Journal of Aviation Psychology*, 9(3):241–269, 1999.

[4] Jens Rasmussen, Annelise Mark Pejtersen, and L.P. Goodstein. *Cognitive Systems Engineering*. John Wiley & Sons Inc, 1994.

[5] Kim J. Vicente. *Cognitive Work Analysis*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1999.

[6] Kim J. Vicente and Jens Rasmussen. Ecological interface design: Theoretical foundations. *IEEE Transactions on Systems, Man and Cybernetics*, 22(4):589–606, July/August 1992.

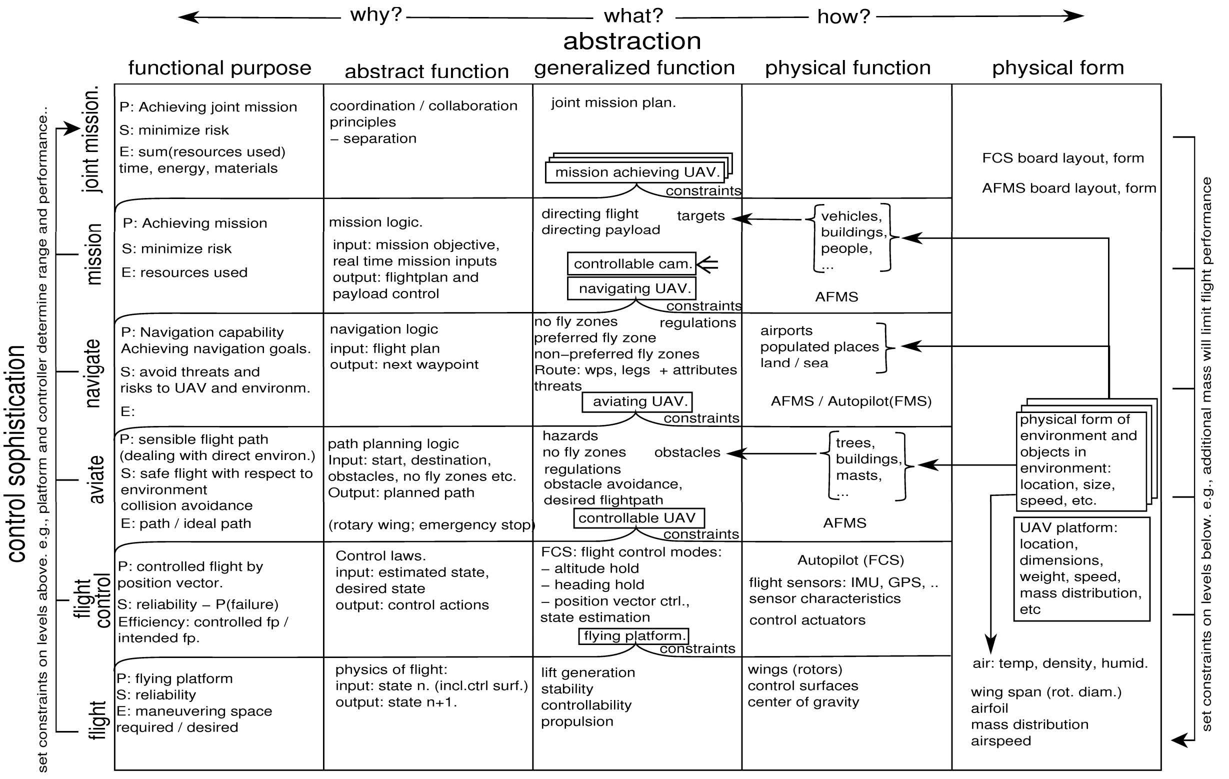

Figure 4: Abstraction – Sophistication Analysis for UAV control. With Production(P), Safety(S), and Efficiency(E) defined at the functional purpose level at each level of sophistication.

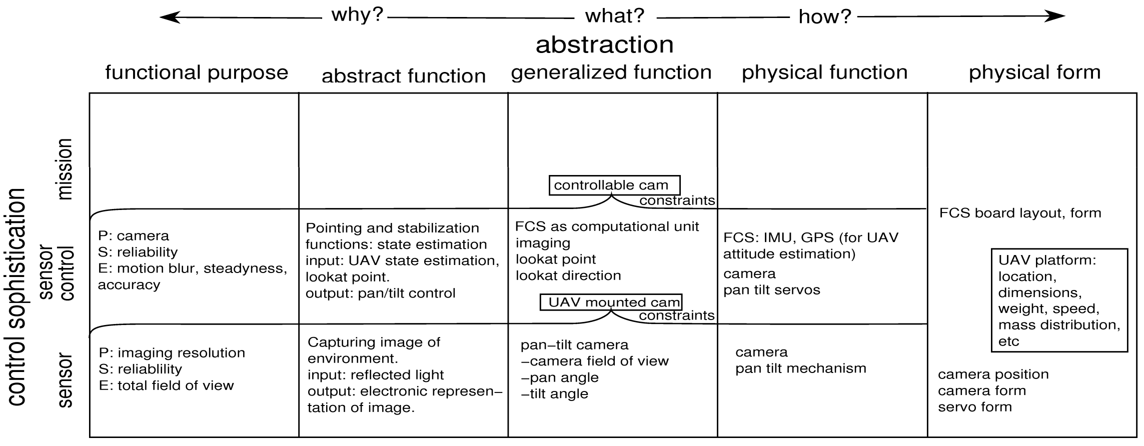

Figure 5: Abstraction – Sophistication Analysis for UAV payload control.