A HIGH LEVEL ACTIVE PERCEPTION CONCEPT FOR UAV MISSION SCENARIOS

C. Hellert[1], D. Smirnov[1], M. Russ[1], P. Stütz[1]
[1]Universität der Bundeswehr München, Neubiberg, Germany

Abstract

Many robotic systems use active perception methods to sense the environment and execute specific tasks with pre-defined signal processing chains. The paradigm of active perception tries to enhance perceptual performance by dynamically interacting with the signal processing chain and their containing algorithms. In this article we propose a concept that uses such active perception techniques to perform mission relevant tasks onboard UAVs. The concept utilizes a resource base containing information on available active resources such as computer vision algorithms, hardware resources, available sensors and actuators as well as background knowledge such as previously trained models or geographic information. This resource base also includes attributes on applicability, capability and limitation of each item. Our idea is to generate a signal processing chain dynamically dependent on current mission needs. This is achieved by understanding which resources are likely to fulfill a specific mission task. Thereby specific focus shall be given to effectively balancing and distributing the resulting processes to be performed on available hardware resources, while taking into account limited processing power and energy supply.

1. MOTIVATION

In recent years development and research progress towards highly automated systems has increased tremendously in the field of mini and micro UAVs [1] [2] [3]. Automation techniques for platform stabilization, guidance and control have been intensively investigated and developed in the past decades and are broadly applicable in the unmanned flight arena. However, when it comes to providing automation functions beyond mere platform locomotion, taking into account the actual intended purpose of the flight mission, only few approaches have been proposed. Such automation on mission management level has to understand mission goals, plan the course of further actions in various domains of platform and payload handling and be able to assess its own effectiveness [4].

On this level, also sensors will have to provide the necessary feedback on situational settings. The challenge in automating such perception capabilities lies in the broad spectrum of possible mission tasks. Our research effort in the frame of the SAGITTA research program [5] initialized by Cassidian focuses on airborne active perception techniques seeking to provide skills for the “seeing and understanding” of missions relevant cues to the *Missions Management System* (MMS) of an UAV.

FIG 1 depicts the general automation scheme capable to perform mission relevant tasks in a more automated fashion. The human operator is still involved, however mainly provides high level mission tasks to the MMS. Information derived from the *Mission Sensors* are analyzed and attributed by the *Sensor and Perception Management System* and fed into the MMS and/or further to the operator. The general idea behind this approach is to provide automation mechanisms to the operator relieving him from excessive workload by dispatching complex tasks to the on-board system or saving mental resources e.g. to control multiple UAVs [6].

This paper concerns an appropriate concept for such *Sensor and Perception Management* in the field of reconnaissance and surveillance missions along the paradigm of active perception. Thereby the focus is on a task-related vision sensor system utilizing machine resource and capability knowledge. We define that resources are available system elements such as processing units, sensors, actuators as well as image processing algorithms. Knowledge in this context describes features, capabilities and limitations of such resources. Our approach is based on the general concept proposed by Russ and Stütz [7] and adapted to our research project.

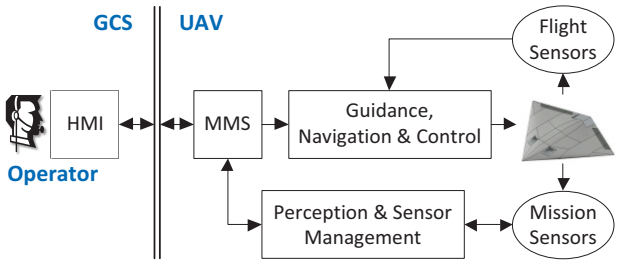

FIG 1. Concept for automation of an UAV through a *Perception and Sensor Management* System.

The paper is structured as follows: In Section 2 we give a brief summary of related research and other projects. The SAGITTA demonstrator and *Perception and Sensor Management* elements are introduced in Section 3. The general system concept is proposed in Section 4 and shall give a basic idea of the interactions and concepts for a realization of the *Perception and Sensor Management* functions. Section 5 will then describe the system concepts in-depth. Section 6 focuses on a simulation environment, especially for the purpose of knowledge generation which is discussed in Section 7. Finally, we give an outlook to further work in Section 8.

2. RELATED WORK

The idea of active perception [8] also known as active vision [9] or image understanding [10] emerged in the last 30 years. The general idea is to model the resources, especially the sensors, and design control strategies to fulfill determined tasks of a vision system using resources. There are several systems that implemented active perception techniques for various applications. Clouard et al. [11] proposed a concept for automatic generation of image processing applications using image processing libraries to construct image processing chains. Another concept is the knowledge-based approach by Matsuyama [12] [13] describing a blackboard where the results from various image processing algorithms are published and validated by a control system. A survey about content-based image retrieval systems is proposed by Liu et al. [14] concerning the description and results of image processing techniques for a semantic level.

To realize active perception systems on UAVs computer vision algorithms for areal images are needed. Commonly, the interests are in global segmentation [15], road segmentation [16] [17] [18], object detection [19] [20] [21] and tracking [22]. Since classification algorithms require knowledge about the object appearance there is a need of supervised or unsupervised learning methods [23]. In addition, sensor planning [24] is an important issue when managing the available sensors by parameters (gain, shutter times, zoom levels, etc.) and positioning.

Real flight tests are very time consuming due to preparation time and analysis processes. In addition, flight tests have to be repeated often because of missing data or evaluation purposes especially for perceptual systems. Therefore, simulation environments [25] can be used before flight tests for performance evaluation. Hummel and Stütz [26] proposed a generic simulation environment for vision sensors using a serious gaming engine where complex missions can be modeled.

3. TARGET APPLICATION

This section is related to the SAGITTA project. It gives an insight into the SAGITTA demonstrator and describes the mission scenario where the concept of our high level active perception system shall be demonstrated. Furthermore it contains information about the mission sensors and the gimbal configuration. The last section gives a brief overview of the *Mission Management Computer* (MMC), its hardware components and the MMC architecture.

3.1. SAGITTA demonstrator

The joint national initiative “Open Innovation / SAGITTA” was founded in 2010 by Cassidian, to respond to the growing need to future unmanned aircraft systems (UAS) and enhance the necessary technological skills. This cooperative approach, in collaboration with renowned German partners from industry, research and science aims to identify and evaluate relevant technologies, to enhance them and to integrate them as far as possible in the fixed-wing flying demonstrator SAGITTA (see FIG 2).

The common goal of the “Open Innovation” initiative is to promote research activities in the “unmanned flying” scope through sustained cooperation in the relevant core technology areas. Our research effort in SAGITTA is to design an appropriate *Sensor and Perception Management* System providing mission relevant perceptual capability. In addition, we design the needed processing hardware as well as the mission sensors.

FIG 2. Project partners participating in the SAGITTA technology demonstrator program.

3.2. Mission Vignette

The concept presented in this paper will be demonstrated along with other research topics on specified missions. Each mission contains a tactical situation, for example as shown in FIG 3.

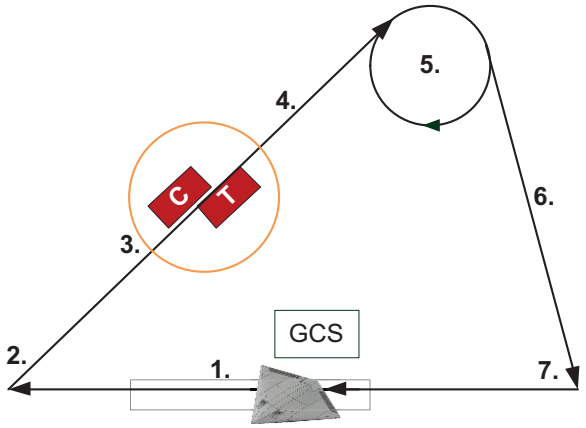

FIG 3. A SAGITTA mission vignette example. The phases are: (1) take-off via departure point, (2) transit to target, (3) perception task “target identification”, (4) task finished, (5) wait for next task, (6) fuel critical, (7) land.

In the given example our research will be demonstrated during the perception task phase “target identification” which is defined as follows: There are two moving vehicles on a road, an ambulance and an armored vehicle. In FIG 3 the ambulance is marked with C (civil) and the armored vehicle is marked with T (target). The task is to distinguish between the two vehicles and track the target. Therefore, the perception tasks can be broken down into:

- Road segmentation
- Vehicle detection
- Vehicle identification
- Tracking and labeling

In addition, to show the interaction with the operator, the identification of the target can be alternatively assisted through the operator by transmiting the images of the detected vehicles to the ground control station (GCS). The definition of the tasks include not only “what” has to be

done, but “how” the tasks have to be solved. This is the focus of the active perception concept that will be proposed in this paper.

3.3. Mission Sensors and Gimbal

We use three different sensors which are an electro-optical (EO) camera, an infrared (IR) camera and a laser range finder. To deploy the SAGITTA sensor suite as flexible as possible, the sensors are not firmly attached to the UAV, but integrated into a gimbal which results in three major advantages: The sensors can be aligned independently of the attitude of the UAV, sensors can be locked to view a distinct location, and the tracking of targets is made possible. In addition, the arrangement of the sensors within the gimbal is equal allowing image registration. The gimbal also provides a dedicated rotational attitude to protect the sensitive equipment against stone chipping during take-off and landing.

The decision to use a complementary set of infrared and electro-optical sensors was based on the ability to scan a large spectral range for day and night operation. Considering a flight altitude of about 1500 meters, a zoom able sensor is needed to get a more detailed view on Regions and Points of Interests (ROIs, POIs) while maintaining the ability to scan large areas. This sensor combination has the following advantages:

- Heat signatures, detected by the IR sensor, can be used to generate ROIs and POIs and further examined by the high resolution zoom-lens EO sensor.
- Since the distance between target objects and the UAV varies, the zoom-lens can preserve a constant scale of the object making identification easier.
- The laser range finder can assist the sensor suite by determining the distance to an object for identification and registration issues.

All sensors and the gimbal are controlled by the *Sensor and Perception Management System*.

3.4. Mission Management Computer

The Mission Management Computer (MMC) onboard of SAGITTA is the processing hardware for the MMS and the *Sensor and Perception Management System*. Because weight, size and power consumption requirements play a significant role we use an embedded CompactPCI system with single board computers communicating via a Gigabit Ethernet (GigE) switch. FIG 4 shows a simplified communication concept of the MMC. There are four major MMC components:

- Decision Engine (DE): All general tasks commanded by the operator via the MDL (Mission Data Link) are interpreted by the DE resulting in a mission plan. This plan includes perception tasks for the *Sensor and Perception Management System*.
- Mission Safety Shell (MSS): The MSS serves as a gateway ensuring that data sent to the *Flight Management System* (FMS) is not threatening SAGITTA e.g. by leaving the mission area.
- Active Perception (AP): Our *Sensor and Perception Management System* is running on the AP board having direct access to the gimbal control unit.
- Hardware Acceleration (HA): This board contains a Field Programmable Gate Array (FPGA) to accelerate image processing algorithm and utilizing the EO stereo system for an automated landing process.

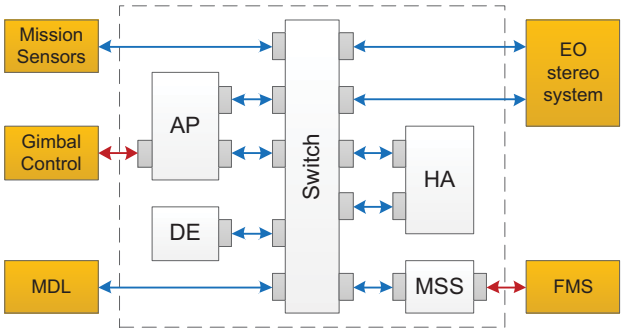

FIG 4. MMC communication overview. Connections colored blue and red are GigE connections and serial interfaces respectively and realized through a custom backplane. All elements within the dashed box are MMC components.

All components are installed in a conduction cooled housing making it robust to shock and vibrations as well as resistant against high humidity.

4. OVERALL CONCEPT

In this section we present our general approach on how to use available resources onboard a UAV to solve given perception tasks. Therefore, the fundamental idea and the related system architecture are described.

4.1. Fundamental idea

An UAV platform is limited in payload such as sensors and computing resources due to restricted energy supply and weight limitations. Therefore it is difficult to add resources or to change the current payload configuration for every mission task that may arise. Hence our approach is to analyze the given UAV platform and derive a machine understanding about the perceptual capabilities in dependency of the available resources. In addition, this supports the case that if some resource became inoperative during mission time, the perception capabilities will then automatically be rearranged. Resources in this context are not only hardware but also software components, which in our case are computer vision algorithms and are modeled as *Perception Modules* (*PM*s).

FIG 5 depicts the structure of an example PM which in this case models a simple processing operator with adjustable parameters. The *Description Interface* provides access to capabilities, domain and environmental constraints, and requirements towards other resources stored in the *Knowledge Database*. Adjusting parameters and observing the status of the PM can be achieved via the *Control Interface*. Each PM has an input and an output descriptor of various types (images, feature descriptors, interest points, etc.). The *Processing Core* contains the implementation of a computer vision algorithm and is controlled via parameters (*Controller*).

For example, a Gaussian convolution

$$(1)\quad f_{(x,y)} * g_{(x,y)} \quad with \quad g_{(x,y)} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

of an image f with a filter mask g expects an image as input and delivers an image as output. The parameters are the *kernel width* and the *standard deviation* σ. The module description contains the following:

- The perception capabilities are *noise reduction* and *blurring*.
- It is independent from domain and environmental constraints.
- The resource requirements are the *input image*, the *memory consumption*, the *processing time* and the *necessary control parameters*. There might be specific parameter sets for distinct cases like specific weather conditions or determined viewports.

The granularity of the PMs has to be determined depending on the complexity of the required PM management and the utilization of the PMs. In the example of the Gaussian convolution a finer granularity can be a general convolution and a large-grained PM can be an image enhancement module. The level of granularity has the following consequences:

- A finer granularity grants more flexibility for parameter adjustments and improves the perceptual capabilities but might also increase the complexity to manage the PMs.
- Large-grained PMs can exclude simple and light-weight image processing methods. It is also possible that the number of parameters rises for one PM and therefore the complexity increases. An advantage is that the system only needs general computer vision knowledge to interact with the PMs.

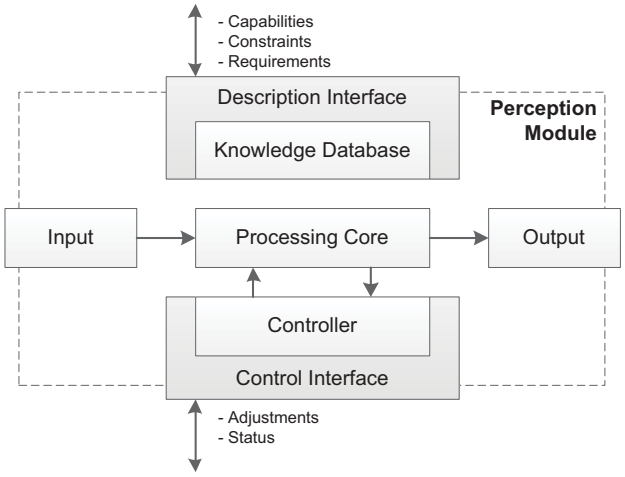

FIG 5. *Perception Module* structure.

Since several PMs have to be combined to fulfill a perception task (typical an object detection) it will be even more complicated finding the appropriate parameters for each PM within the processing chain. To overcome the complexity we will use *Perception Strategies* defined through expert knowledge. A *Perception Strategy* contains the information on what types of PMs are needed and in which order the PMs can be processed in general. The Perception Strategies are represented in a template form where for example the location of interest can be an airbase, a certain road or a village. Furthermore, there might be predefined parameter sets for various environmental conditions for different *Perception Modules*, but we exclude this in the example depicted in FIG 6.

Finally, the application of such *Perception Strategies* on the available PMs will result in *Perception Graphs*. A *Perception Graph* contains all possible *Perception Module* combinations solving a perception task with definite template parameters as shown in FIG 7.

```
TASK DETECTION <object> ON <location>;
DO stabilization;
DO segmentation(location);
DO feature_extraction(object, location);
DO classification(object, location);
```

FIG 6. Example of a *Perception Strategy* for the detection of an object at a determined location. Words highlighted bold are syntax while italic words are template parameters.

Since the *Perception Strategies* are general knowledge available before mission, the related graphs can be generated during mission preparation by the *Perception Solver* (see Section 5). If a specific task is executed during a mission, like detecting a vehicle on a road, the related *Perception Graph* can be adjusted accordingly to the template parameters. An example of such an adjusted graph is shown in FIG 7 based on the *Perception Strategy* shown in FIG 6 where the template parameters are the *object* and *location* type.

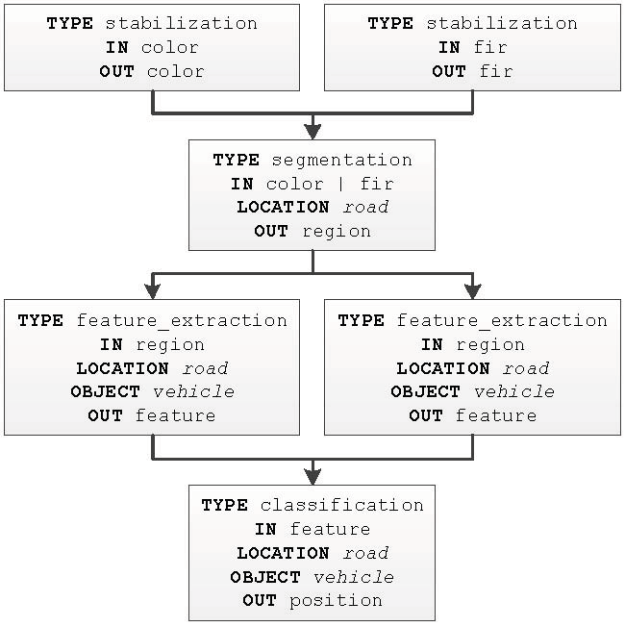

FIG 7. Example *Perception Graph* for detecting vehicles on roads. With sufficient resources four possible processing chains are available for the task of vehicle detection on a road. Thereby, both feature extraction PMs having the same inputs and output but containing different algorithms.

Since a graph can have multiple valid solutions it is important to know which resources are needed for each possible solution. We define that the terminology of a solution is a processing chain. Depending on available resources one or more processing chains can be selected for execution.

4.2. System overview

In the following section we propose a general architecture for a *Sensor and Perception Management System* capable to construct and execute such processing chains as

described above, and taking into account the following design issues:

– Our system shall receive a list of dedicated *Perception Tasks* which are coordinated in time and space.
– The UAV platform itself also shall be considered as a resource in terms of sensor positioning and locomotion. However, contrary to the resources mentioned above, this resource cannot be controlled solely by our system due to competing demands by other mission relevant subsystems. Therefore, we make arrangements to work around this iteratively by *platform requests* and feedbacks (see Section 5).
– The system shall configure its components dependent on the variable surveillance tasks during mission time.
– To allow an adaption to different platforms, the system shall be independent from the resources. This means, that the available resources shall be recognized by the system defining its capabilities and limitations and therefore making it available for various, different platforms (e.g. fixed-wing or helicopter).

On top level we divide the system into three layers (FIG 8). The lowest layer comprises the resources controlled and managed by the *Sensor Management* layer. The *Perception Management* layer communicates with the *Mission Management* and uses the *Sensor Management* to fulfill *Perception Tasks*. The *Mission Management* will be informed about the capabilities of the *Sensor and Perception Management* to be able to plan the mission tasks properly. Relating to the presented fundamental idea a typical process is as follows: When a *Perception Task* is sent to the *Perception Management* by the *Mission Management,* an appropriate *Perception Graph* will be selected and the resource requirements are proposed to the *Sensor Management*. The *Sensor Management* will verify the requirements dependent on the available resources and commit the feasibility to the *Perception Management*. Finally, the *Perception Management* will select and execute feasible processing chains and inform the *Mission Management* about the observation measurements. In addition, it might be possible that a *Perception Task* can only be fulfilled when the platform changes the attitude or altitude which will be committed via platform requests.

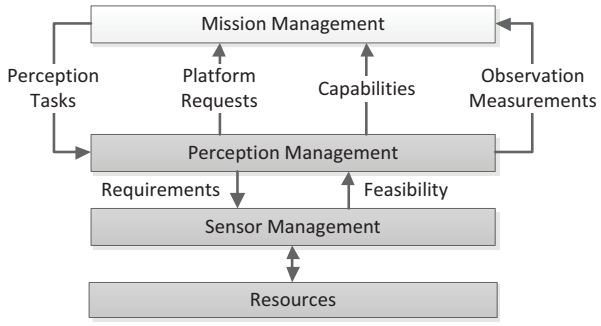

FIG 8. Overall system layout. The *Mission Management* defines a top level system providing *Perception Tasks* to our system.

5. DETAIL ARCHITECTURE DESCRIPTION

In this section a more detailed description of the *Sensor and Perception Management System* architecture as shown in FIG 9 is presented.

5.1. Perception graph analysis

As introduced in Section 4 we use *Perception Modules* and *Perception Strategies* to generate *Perception Graphs* before flight. A *Perception Knowledge* database can be accessed by the *Perception Solver* containing the pre-generated *Perception Graphs*. In addition, the *Perception Strategies* are also available within the *Perception Knowledge* database for dedicated parameter sets.

If a *Perception Task*, sent from the *Mission Management System*, is received by the *Perception Solver*, the resource requirements are handed over to the *Sensor Planner*. The graph comprises the resource requirements through the inputs, outputs, parameters and configurations of the involved *Perception Modules*. Therefore, the *Sensor Planner* checks the requirements based on actual available resources. The following dependencies are considered:

– The *Sensor Configurator* manages and grants access to available sensors. Hence, the *Sensor Planner* can determine available sensor resources through the *Sensor Configurator*. In addition, the *Sensor Configurator* can adjust the sensor parameters dependent on domain and environmental conditions.
– Needed processing requirements are computed for each *Perception Module* within the actual *Perception Graph* by using memory and processing time information provided by the *Module Controller.* Current free processing resources are monitored by the *Processing Monitor* making it possible for the *Sensor Planner* to determine if a certain processing chain can be executed.
– If a certain *Perception Module* carries platform dependent requirements, e.g. specific flight altitude, the *Sensor Planner* issues a platform request to the *Mission Management* layer.
– In the most cases the sensors must be positioned for tracking and identification tasks. Therefore, the *Gaze Control* commands the gimbal and the *Sensor Planner* can retrieve the current state to check gimbal related requirements.
– Knowledge about the domain (e.g. terrain) may be used for image processing tasks. Hence, topographic and topologic information shall be available and will be stored in the *Domain Knowledge* database.

After the resource requirements have been analyzed by the *Sensor Planner* the results are committed to the *Processing Coordinator* determining possible processing chains.

5.2. Resource Virtualization

To allow a platform independent design the resources need a general control and data interface. Therefore, the virtualization of the resources addresses the following issues:

– All required data are stored in short-term buffers including sensor and gimbal data, as well as flight parameters. Therefore it is no longer necessary to acquire needed data from each resource directly and it supports algorithms requiring data of the recent past, e.g. for background subtraction.
– It is possible for more than one processing chain to be executed in parallel to improve results. Therefore, data has to be interchanged by multiple *Perception*

Modules. Such data can be also stored within buffers of the *Resource Virtualization* stage.

– Since the scenery to observe is often dynamical, sensor parameters and the direction of the sensors must be changed during perception chain execution. A typical task would be the tracking of a vehicle. Such adjustments can be done via a control interface from the *Processing Coordinator* to the *Resource Virtualization* stage.
– Considering a distributed computer system, the *Resource Virtualization* will also be used to synchronize and distribute data.

In conclusion, the *Resource Virtualization* is an encapsulation of the available resources and a short-term data storage containing the data of recent times and providing a generic control interface to the available resources. In addition it provides interprocess communication (IPC) abilities.

5.3. Process execution and validation

As described above, the *Processing Coordinator* has access to the *Perception Graph* and the results of the verified resource requirements of the *Sensor Planner*. In general, there are three cases that can occur during the selection of the appropriate processing chains:

– The simplest case is when only one valid processing chain is available for solving the *Perception Task*. Hence, this processing chain is executed when sufficient resources are available or can be adjusted in processing performance (e.g. reducing image resolution).
– When more than one processing chain is available through the graph, then the *Processing Coordinator* will compute the needed resources for each processing chain p_c and selects the optimal solution. The optimal solution is a minimum search of the cost function as shown in Formula (2), where $c = \{1,...,M\}$ and M is the number of processing chains. The number of *Processing Modules* is donated by N_c within the processing chain p_c. Hence, $m_{c,i}$ is the memory consumption of the *Perception Module* i in processing chain p_c and m is the total amount of available memory. The processing time consumption of a *Perception Module* is denoted by $t_{c,i}$. In addition, remaining energy of the power supply can be taken into account as well as real-time issues.
– In the case that are enough resources available, it is possible, to execute more than one processing chain.

$$C = \arg\min_{c}\left(\sum_{i=1}^{N_c}\frac{m_{c,i}}{m} + t_{c,i}\right) \quad (2)$$

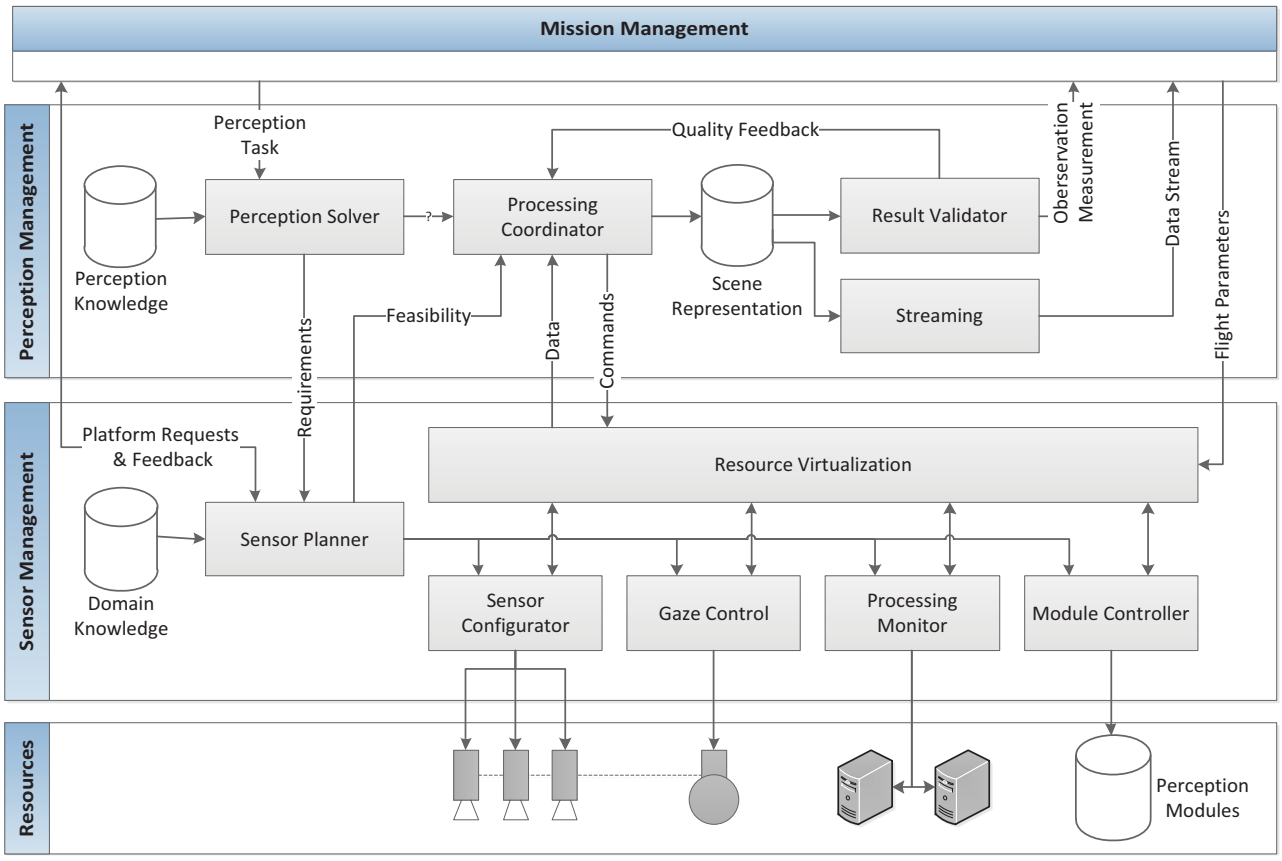

FIG 9. System architecture for *Sensor and Perception Management*.

Since image processing algorithms are not entirely reliable it is good practice to validate their results. In addition, regarding the last case above, the results have to be checked for consistency. Therefore, the *Result Validator* observes the *Scene Representation* containing the recent results of one or more processing chains, and represents the actual understanding of the environment. The first step is a plausibility check examining the results with knowledge about the target object or the environment. In general, the following issues are resolved by the plausibility check:

– If an object shows an unusual behavior detected by using motion models, the result can be removed or corrected by probability calculations. For example, if a vehicle is tracked by the system and the vehicles trajectory suddenly changes, violating the maximum acceleration or the minimum turning radius, then the result might be discarded.
– Superposition, in conjunction with a geographical map and segmentation results, can help to find false detections (e.g. a vehicle is detected on a house). The solution can be to discard or correct the result, for example by moving the location of the detected vehicle to a close road segment.
– When additional information about the environment are detected, for example if the *Perception Task* is to find a moving vehicle and the system detects a parked vehicle, there a two possible solutions: The result can be ignored or committed to the MMS to enable reaction on unpredicted events.

If there is more than one processing chain concerning the same task a consistency check have to be performed. For example if three processing chains are running concurrently, and two processing chains find a vehicle on the same location and one does not, then the *Result Validator* will discard the result from the processing chain finding no vehicle. This can be realized by data fusion for example with a Kalman filter. When there are only two processing chains having varying results then there are several solutions:

– The result from the more trustable processing chain can be retained, but the problem is which processing chain has the highest accuracy. For example, this can be solved when using classification algorithms [23] evaluated on the same dataset. The calculated detection rate can then be used as a credibility factor and may also be included into the cost function (see Equation (2)) to balance processing requirements and accuracy. In addition, precision evaluations from literature can be interpreted as a credibility factors, e.g. for local descriptors [27].
– The most image processing algorithms have parameters (thresholds, filter masks, learning rates) [28]. Therefore, parameter sets can be defined for various situations. Dependent on the current results in the *Scene Representation* the parameter sets can be changed by the *Processing Coordinator* using the quality feedback from the *Result Validator*.
– The simplest solution is to discard both results and wait for new measurements which are even. Asking the operator for advice is an alternative approach. Therefore the *Streaming* module provides the ability to transmit images, videos or any further information to the operator.

Finally, the results of the consistency and plausibility check are combined to a quality feedback for the *Processing Coordinator* to adjust the processing chains. In addition proven results will be sent to the *Mission Management* as *Observation Measurements*.

6. EVALUATION AND SIMULATION ENVIRONMENT

Due to cost and time factors it is very costly to develop and evaluate the concepts and algorithms in all aspects during real flights. Therefore a suitable simulation environment for testing close to the mission vignette is necessary. Such simulation environment would need to model and animate the mission scenario and to include the emulation of all mission sensors and the gimbal, as well as the flight dynamics of the SAGITTA platform.

We selected the Virtual Battle Space 2 (VBS2) [29] engine as the basis of our simulation environment. VBS2 brings along a realistic graphical representation for EO and IR sensors, sophisticated physics models, and a very wide range of supporting assets (textures, objects, decals, etc.) for modeling different scenarios. VBS2 allows a detailed modeling of imaging sensors including features such as opening angle, shutter speed or zoom-lens, as well as adding noise or distortions to realize realistic sensors. In addition the simulation of a gimbal is already integrated and can be adjusted to the dynamics of ours.

The VBS2 engine shall not only be used to generate synthetic scenarios, but also to test the implemented image processing algorithms under different conditions. Therefore a specific scenario can be modified with different atmospheric and lighting conditions to gain insight in appropriate PMs and the optimal parameter settings for each PM. The VBS2 engine provides a realistic graphical simulation shown in FIG 10. Hence, the obtained knowledge (parameter sets, capabilities and limitations) can be also used on real video streams.

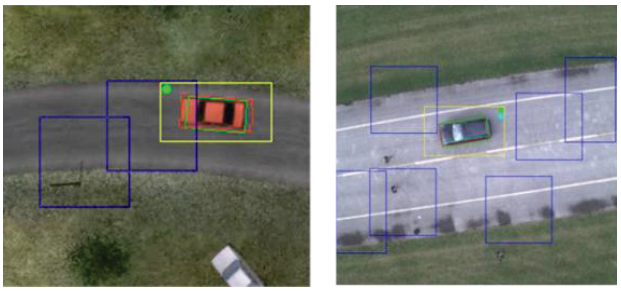

FIG 10. Vehicle detection in a simulation and real image.

FIG 11 shows the concept of the evaluation and simulation framework. The simulation environment consists of a VBS2 server and two VBS2 clients. The Server manages the Flight Management System (FMS), the flight dynamic and the auto flight system of the UAV. It is also responsible for the graphical simulation environment and the communication between the two clients. The camera sensors are simulated with the VBS2 clients, where one client is responsible for the EO sensor and the other client emulates the IR sensor and the gimbal. Positional alterations of the gimbal will affect both sensors. For the SAGITTA demonstrator only sensors with GigE Vision [30] interfaces have been selected. Therefore, the video streams are converted and preprocessed into a GigE Vision virtual sensor, abstracting simulated sensors and real hardware. Thus, Hardware in the Loop Simulation

(HILS) without modifying software components or using other hardware is achieved.

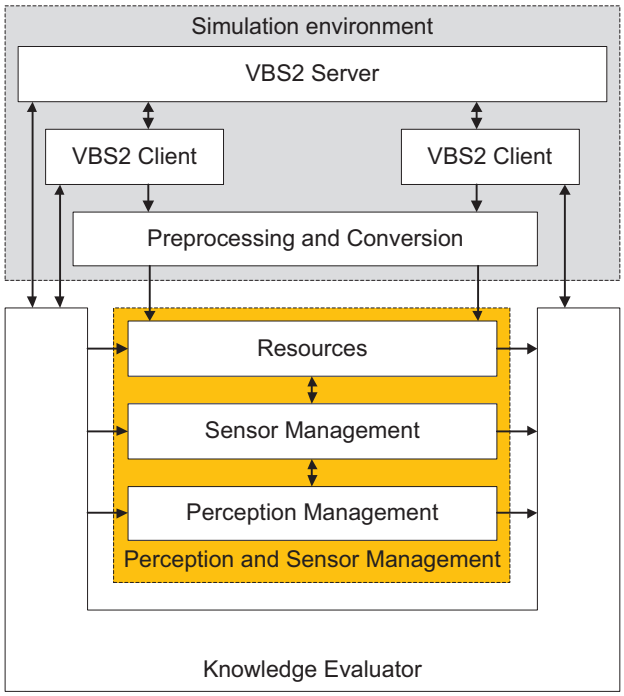

FIG 11. Architecture of the simulation and evaluation framework.

The evaluation environment consists of a *Knowledge Evaluator* where our *Sensor and Perception Management* concept is embedded. The *Knowledge Evaluator* is responsible for the following tasks:

– It shall send a *Perception Task* to *Sensor and Perception Management* while starting an appropriated mission, and compare the results from the simulation (ground truth) and the system to evaluate the quality.
– The *Knowledge Evaluator* shall be able to test different processing chains or single PMs for determining the limitations and capabilities.

In conclusion, the *Knowledge Evaluator* collects the results from the *Sensor and Perception Management* and tries to enhance the performance resulting in knowledge about the PMs.

7. KNOWLEDGE GENERATION

In this section we address the problem of how the knowledge about a certain *Perception Module* is gained. There is a knowledge database in each PM as described in Section 4. This database contains two major items which are the description of the capabilities and limitations as well as required trained models. The capabilities and limitations of a PM can be obtained by the following:

– Since a PM contains a computer vision algorithm, the developer or the publisher of that algorithm knows about its capabilities and limitations and can therefore describe the PM. This is also known as expert knowledge.
– There is literature that evaluates and compares algorithms under various circumstances which can be used to describe additional abilities, as well as using the comparison to add a credibility factor, helping the system to select the most likely result.
– The proposed simulation environment shall be further used to test a PM under various conditions to determine its capabilities and limitations as well as appropriate parameter sets for several situations as described in Section 6.

The first issues must be done manually while the last point is a more automated way to generate the description of a PM. The training of the required models must be also included into the knowledge database and affects the capabilities and limitations of the PM.

In the following, the process of knowledge generation will be explained with the help of a model-based vehicle classification PM. The first step is the training of the model. In general, this can be achieved by the following methods:

– The common and most simple approach is to use an existing database for vehicle classification to generate the model.
– The generation of synthetic databases with the proposed simulation environment requires more effort but can be automated. In addition, there is a risk that the synthetic databases are insufficient for real data.
– The manual creation of a database with recorded flight data is possible but requires higher effort and shall be avoided.

If a database exists, a supervised training of the model can be done with several parameters resulting in multiple models which must be evaluated and the most appropriate models are then included into the knowledge database of the PM. The next step is to create or use a scenario where a vehicle shall be detected. This scenario should have various environmental options, like weather conditions, different vehicles and varying landscapes. The environmental options shall be defined by the specific application to reduce the effort. With the help of the simulation environment it is possible to evaluate the performance of the classification algorithm by comparing the predicted results and the ground truth from the simulation. By varying the environmental options and parameters of the PM, the capabilities and limitations can be extracted. For the example of the classification algorithm, the result will be a matrix containing the environmental options, parameters and the related detection rates, which can then be used to generate the knowledge about the PM.

8. CONCLUSION AND FUTURE WORK

In this paper we presented an active perception concept for UAV mission scenarios and disused its requirements and advantages. Since the concept shall be demonstrated on the SAGITTA demonstrator the mission sensors, the processing hardware and the mission scenario are described. Afterwards a deeper insight into the concept and the related architecture is proposed. Finally, we discussed the advantages and the utilization of a simulation environment and how the required knowledge databases are generated through the evaluation framework.

In order to achieve a first implementation of the proposed *Sensor and Perception Management* concept, we will focus in the next steps on relevant functions to simulate the mission vignette and to evaluate our system to reveal

the possible limitations and enhancements of our proposed concept. Since our system is dependent on knowledge databases it is important to find a suitable knowledge representation language and a proper way to generate the knowledge. Hence the implementation and evaluation of image processing algorithms concerning the mission vignette will be used to find suitable knowledge descriptions.

REFERENCES

[1] G. Granlund, K. Nordberg, J. Wiklund, P. Doherty, E. Skarman, and E. Sandewall, *WITAS: An Intelligent Autonomous Aircraft Using Active Vision*, Proceedings of the UAV 2000 International Technical Conference and Exhibition, 2000.

[2] A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, I.V. Perez, L.G. Gutierrez, D.X. Viegas, M.A.G. Benitez, A. Mallet, R. Alami, R. Chatila, G. Hommel, F.C. Lechuga, B. Arrue, J. Ferruz, J.R.M. Dios, and F. Caballero, *Multiple eyes in the skies - Architecture and perception issues in the comets unmanned air vehicles project*, IEEE Robotics & Automation Magazine, vol. 12, no. 2, pp. 46–57, 2005.

[3] B. Sinopoli, M. Micheli, G. Donato, and T. Koo, *Vision based navigation for an unmanned aerial vehicle*: IEEE International Conference on Robotics and Automation (ICRA), 2001.

[4] J. Uhrmann and A. *Schulte, Concept, Design and Evaluation of Cognitive Task-based UAV Guidance*, *International Journal on Advances in Intelligent Systems*, vol. 5, no. 1 & 2, pp. 145–158, 2012.

[5] H. Williams, *Cassidian targets Sagitta at future UAV designs*: IHS Jane's. Available: http://www.janes.com (2012, Aug. 03).

[6] S. R. Dixon, C. D. Wickens, and D. Chang, *Mission Control of Multiple Unmanned Aerial Vehicles: A Workload Analysis*, Journal of the Human Factors and Ergonomics Society, vol. 47, no. 3, pp. 479–487, 2005.

[7] M. Russ and P. Stuetz, *Airborne sensor and perception management: A conceptual approach for surveillance UAS*, in-press.

[8] R. Bajcsy, *Active perception*, Proceedings of the IEEE, vol. 76, no. 8, pp. 966–1005, 1988.

[9] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, *Active vision*, International Journal of Computer Vision, vol. 1, no. 4, pp. 333–356, 1988.

[10] D. Crevier and R. Lepage, *Knowledge-Based Image Understanding Systems: A Survey*, Computer Vision and Image Understanding, vol. 67, no. 2, pp. 161–185, 1997.

[11] R. Clouard, A. Elmoataz, C. Porquet, and M. Revenu, *Borg: a knowledge-based system for automatic generation of image processing programs*, IEEE Trans. Pattern Anal. Machine Intell, vol. 21, no. 2, pp. 128–144, 1999.

[12] T. Matsuyama, *Expert systems for image processing-knowledge-based composition of image analysis processes*, Computer Vision, Graphics, and Image Processing, vol. 48, no. 1, pp. 22–49, 1988.

[13] T. Matsuyama, *Knowledge-Based Aerial Image Understanding Systems and Expert Systems for Image Processing*, IEEE Transactions on Geoscience and Remote Sensing, vol. 25, no. 3, pp. 305–316, 1987.

[14] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, *A survey of content-based image retrieval with high-level semantics*, Pattern Recognition, vol. 40, no. 1, pp. 262–282, 2007.

[15] H. Permuter, J. Francos, and I. *Jermyn, A study of Gaussian mixture models of color and texture features for image classification and segmentation: Graph-based Representations*, *Pattern Recognition*, vol. 39, no. 4, pp. 695–706, 2006.

[16] I. Laptev, H. Mayer, T. Lindeberg, W. Eckstein, C. Steger, and A. Baumgartner, *Automatic extraction of roads from aerial images based on scale space and snakes*, Machine Vision and Applications, vol. 12, no. 1, pp. 23–31, 2000.

[17] E. Frew, T. McGee, Z.W. Kim, X. Xiao, S. Jackson, M. Morimoto, S. Rathinam, J. Padial, and R. Sengupta, *Vision-based road-following using a small autonomous aircraft,* Aerospace Conference, 2004.

[18] S. Rathinam, Zu Kim, A. Soghikian, and R. Sengupta, *Vision Based Following of Locally Linear Structures using an Unmanned Aerial Vehicle*, 44th IEEE Conference on Decision and Control, pp. 6085–6090, 2005.

[19] T. Zhao and R. Nevatia, *Car detection in low resolution aerial images*, Image and Vision Computing, vol. 21, no. 8, pp. 693–703, 2003.

[20] K. Kaaniche, B. Champion, C. Pegard, and P. Vasseur, *A Vision Algorithm for Dynamic Detection of Moving Vehicles with a UAV*, Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA), pp. 1878–1883, 2005.

[21] J. Gleason, A.V. Nefian, X. Bouyssounousse, T. Fong, and G. Bebis, *Vehicle detection from aerial imagery*, International Conference on Robotics and Automation (ICRA), 2011.

[22] V.N. Dobrokhodov, I.I. Kaminer, K.D. Jones, and R. Ghabcheloo, *Vision-based tracking and motion estimation for moving targets using small UAVs*, American Control Conference, 2006.

[23] V. N. Vapnik, *The nature of statistical learning theory,* 2nd ed. New York: Springer, 2000.

[24] K. Tarabanis, P. Allen, and R. Tsai, *A survey of sensor planning in computer vision*, *IEEE Transactions on Robotics and Automation,* vol. 11, no. 1, pp. 86–104, 1995.

[25] J. Craighead, R. Murphy, J. Burke, and B. Goldiez, *A Survey of Commercial & Open Source Unmanned Vehicle Simulators*, International Conference on Robotics and Automation, pp. 852–857, 2007.

[26] G. Hummel and P. Stütz, *Conceptual design of a simulation test bed for ad-hoc sensor networks based on a serious gaming environment*, in International Training and Education Conference, 2011.

[27] K. Mikolajczyk and C. Schmid, *A performance evaluation of local descriptors*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pp. 1615–1630, 2005.

[28] Z. Zhang, *Parameter estimation techniques: a tutorial with application to conic fitting*, Image and Vision Computing, vol. 15, no. 1, pp. 59–76, 1997.

[29] Bohemia Interactive Australia Pty Ltd, *White Paper: Virtual Battle Space 2.* Available: http://distribution.bisimulations.com/docs/VBS2_White paper.pdf (2012, Jul. 26).

[30] AIA, *GigE Vision Standard.* Available: http://www.visiononline.org (2012, Jul. 26).