Development of a behaviour-based control and software architecture for a visually guided mine detection robot

Daniela Doroftei[*,**] **— Eric Colon**[*] **— Yvan Baudoin**[*] **— Hichem Sahli**[**]

[*] *Royal Military Academy, Department of Mechanical Engineering (MSTA)*
Avenue de la Renaissance 30, B1000 Brussels, Belgium
{daniela.doroftei;eric.colon;yvan.baudoin}@rma.ac.be

[**] *Vrije Universiteit Brussel, Department of Electronics and Informatics (ETRO)*
Pleinlaan 2, B1040 Brussels, Belgium
hsahli@etro.vub.ac.be

ABSTRACT. *Humanitarian demining is a labor-intensive and high-risk which could benefit from the development of a humanitarian mine detection robot, capable of scanning a minefield semi-automatically. The design of such an outdoor autonomous robots requires the consideration and integration of multiple aspects: sensing, data fusion, path and motion planning and robot control embedded in a control and software architecture. This paper focuses on three main aspects of the design process: visual sensing using stereo and image motion analysis, design of a behaviour-based control architecture and implementation of a modular software architecture.*

RÉSUMÉ. *: Le déminage humanitaire reste une opération laborieuse et risquée. Dans ce contexte, le but est de développer un robot mobile démineur capable de scanner un champ de mine de manière semi-automatique. La conception d'un tel robot doit prendre en considération différents aspects: la perception et le fusion des données, l' architecture de contrôle pour la navigation du robot et son implémentation avec un logiciel adapté. Le système de perception utilisé dans ce travail est basé sur la stereo vision et l'analyse du mouvement dans l'image. Le système de contrôle utilise une architecture modulaire basée sur les comportements.*

KEYWORDS: *Stereo vision, Image motion analysis, Behaviour-based robot control, Robot control and software architectures, Behaviour fusion, Robots for risky interventions.*

MOTS-CLÉS : *Stereo vision, Traitement de mouvement dans l'image, Contrôle basé sur les comportements, Fusion de comportements, Robots pour interventions à risques.*

1. Introduction

1.1. *Goal and problem formulation*

The goal of this research project is to prepare the ROBUDEM, an outdoor mobile robot platform as shown on Figure 1 for a humanitarian demining application. In this setup, the robot navigates and searches for mines by moving a metal detector, detecting suspicious objects in the soil. Once a suspicious object is detected, the robot stops and invokes its Cartesian scanning mechanism. This scanning mechanism performs a 2D scan of the soil, allowing mine imaging tools to make a reliable classification of the suspicious object as a mine or not. This paper describes partial aspects of this research work aimed at making the robot capable to scan a minefield semi-autonomously and return a map with locations of suspected mines.

During the design of all these sub-aspects, the outdoor nature of the robot has to be taken into account, because outdoor robots face special difficulties compared to their indoor counterparts. These include totally uncontrolled environments, changing illumination, thermal, wind and solar conditions, uneven and tough terrain, rain, etc.

Figure 1. *ROBUDEM robot with the metal detector and a stereo camera*

1.2. *Previous work*

For decades, autonomous robotics is a popular research area, yet the amount of real intelligent autonomous outdoor robots applied on the field is still very limited. The goal of the research project presented here is to develop intelligent autonomous robotic agents which can assist humans for various types of risky outdoor interventions (surveillance, crisis management...). The challenges for such a robotic system are tremendous and span various fields of research: from sensing and sensor fusion to modeling and control, map building and path planning, decision making and autonomy, and to the final integration of all these components. Three of these aspects are investigated more profoundly in this paper: visual sensing, robot control and the software architecture. The following paragraphs give an overview of the different existing algorithms and design choices for these different components.

Robotic agents can rely on numerous types of sensors to gain knowledge about the environment or about itself or its position. These sensors include infrared sensors, ultrasound sensors, laser range scanners, GPS, inertial navigation systems... Cognitive science and biological examples pointed out the importance of visual sensing, which led to the application of computer vision algorithms like stereo vision (Park *et al.*, 2005), obstacle detection (DeSouza *et al.*, 2002), person following (Enescu *et al.*, 2005), visual servoing (Hong *et al.*, 2001) to robotics and it eventually also led to mixed paradigms like visual simultaneous localisation and mapping (VSLAM) (Davison *et al.*, 2007). However, integration of vision modules into a control architecture for an autonomous mobile robot is more difficult than just adding the vision components (Schlegel *et al.*, 2000). This is due to the high bandwidth and processing requirements of vision sensors, which require a task-specific configuration of vision-based behaviors. Another drawback of many computer vision algorithms is that they lack stability and robustness when confronted with varying illumination conditions, as it generally happens in outdoor situations, although illumination-invariant algorithms have been proposed (DeCubber *et al.*, 2004). For visual sensing, the Robudem robot is outfitted with a stereo camera system, consisting of two digital cameras. In order to maximize the information stream towards the navigation unit, two different visual processing techniques are used: stereo vision and image motion analysis.

An autonomous mobile robot must be self-reliant to operate in complex, partially known and challenging environments using its limited physical and computational resources. Its control system must ensure in real time that the robot will achieve its tasks despite all these constraints (Medeiros, 1998). One of the first robot control architectures was the Sense Model Plan Act (SMPA) paradigm. The primary drawback of this approach is that the series of stages through which all sensor data must pass places an unavoidable delay in the loop between sensing and action. This is a major problem with all deliberative approaches: since they rely on a world model, they have difficulties with real-time performance in complex environments. To counter this drawback, alternatives, such as the behavior-based approach, were proposed (Arkin, 1987),(Maes, 1989),(Mataric, 1997),(Brooks, 1986). Contrary to deliberative approaches, these behaviour-based control systems afford modular de-

velopment, real-time robust performance within a changing world and incremental growth. In behavior-based control, the control of a robot is shared between a set of purposive perception-action units, called behaviors (Pirjanian, 1999). Based on selective sensory information, each behavior produces immediate reactions to control the robot with respect to a particular objective, a narrow aspect of the robot's overall task such as obstacle avoidance or wall following. Behaviors with different and possibly incommensurable objectives may produce conflicting actions that are seemingly irreconcilable. Thus a major issue in the design of behavior-based control systems is the formulation of effective mechanisms for coordination of the behaviors' activities into strategies for rational and coherent behavior. This is known as the action selection problem. Numerous action selection mechanisms have been proposed over the last decade; a qualitative overview can be found in (Doroftei, 2006). The behavior-based controller presented here uses statistical reasoning on the output data of each behaviour to determine the stability and reliability and therefore also the activity level of seven behaviours, each proposing a (different) velocity and turning prescript.

Robot control architectures become more and more complex, as human reasoning is mimicked. Moreover, there is a significant portion of robot functionality that is common to a large number of robotic systems in different application domains. Unfortunately, most functionality implementations are tied to specific robot hardware, processing platforms and communication environments. Most research and development in software for robotic systems is based on proprietarily designed architectures invented from scratch. To avoid this, the choice of a flexible, extendable and real-time capable software architecture is crucial. This architecture has to ease the use of reusable and transferable software components. Multiple software architectures, like Orocos (Bruyninckx, 2001), Player/Stage (Gerkey *et al.*, 2003), Campout (Pirjanian *et al.*, 2000), CoRoBa (Colon *et al.*, 2006), etc., have been proposed in the past, all with their strengths and weaknesses. The existence of such a multitude of software frameworks hasn't helped the standardisation of robot software architectures. In the course of this research project, the Modular Controller Architecture (MCA) (Scholl *et al.*, 2002) was employed. MCA is a modular, network transparent and realtime capable framework targetted towards the control of autonomous robots.

This paper is structured as follows: in Section 2 the visual sensing is explained in detail. To maximize the information extracted from the visual camera data, two approaches for visual obstacle detection are followed: classical stereo vision and image motion analysis. Section 3 describes the robot control architecture: first the general architecture is explained, then the metal and mine detection groups are discussed. In Section 3.4, the behaviour-based navigation controller is presented, which ensures semi-autonomous robot navigation, while Section 3.5 discusses how this navigation fits in the global framework of the robot motion scheduler. Section 4 describes the implemented software architecture. First, the underlying MCA framework is introduced, then the implementation of the general control architecture is explained and after that the implementation of the behaviour-based navigation controller in MCA is discussed. Finally, some results of experiments with the presented architecture on a real demining robot are presented and some conclusions are formulated.

2. Visual sensing

2.1. *Stereo vision*

Stereo vision employs the difference in location between two cameras. This difference in space leads to two images where the same points can be found at different positions. The goal of stereo disparity estimation is finding the correct correspondences between image points from the left and right camera. For this, we use the algorithm presented in (Birchfield, 1997). The algorithm matches individual pixels in corresponding scanline pairs while allowing occluded pixels to remain unmatched, then propagates the information between scanlines. The algorithm handles large untextured regions, uses a measure of pixel dissimilarity that is insensitive to image sampling, and prunes bad search nodes to increase the speed of dynamic programming. The output of this algorithm is a dense depth map of the area in front of the cameras, as shown in Figure 2. On the depth map in Figure 2c, nearby objects appear dark. The cross on top marks the location of the closest obstacle, which is the darkest point on the depth map and which corresponds here to the obstacle in front of the robot. The data-content of this dense depth map must now be reduced to be useful for the navigation controller. For this, we use the approach proposed by Schafer in (Schafer *et al.*, 2005). Following this method, the dense depth map is downprojected onto the ground surface, such that it can be represented as a 2D line as shown in Figure 2d. This data is further reduced in dimensionality by calculating from the depth line the distance to the nearest obstacle on the left d_l, in the middle d_c, and on the right d_r.

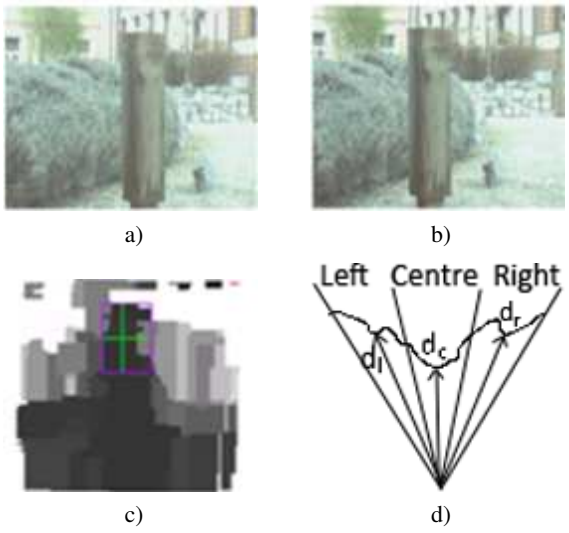

Figure 2. *a) left camera image; b) right camera image; c) dense depth map (white = far, dark = near); d) depth line with nearest distances to obstacles on the left, in the middle and on the right*

2.2. *Image motion analysis*

Motion analysis can provide extra information about the environment. The rationale behind the usage of the image motion for navigation purposes is that when large image motion is detected, this is likely due to objects close to the camera (and thus close to the robot), which requires an appropriate reaction from the obstacle avoidance module. On the other hand, when few image motion is detected, this means that the way in front of the camera is probably quite clear of obstacles.

Multiple techniques stand at our disposal to estimate the image motion. These methods differ in their approach according to the main problem to be solved in image motion: the background estimation and subtraction process. As the camera system is installed on a moving robot system, background estimation is particularly difficult in this case, as it is very hard to build up a model of the background over a large amount of time. This constraint limits the use of traditional advanced background estimation techniques like kernel density estimators, mean shift or mixtures of Gaussians (Piccardi, 2004). As a result, the frame difference between successive frames was employed to find back the moving objects. As expressed by Equation [1], the motion m_k for each pixel is robustly estimated by calculating the frame difference when the difference is above a certain threshold which is dependent on the robot velocity V.

$$m_k = \begin{cases} 0 & \text{if } |frame_i - frame_{i-1}| < cV \\ |frame_i - frame_{i-1}| & \text{if } |frame_i - frame_{i-1}| > cV \end{cases} \quad [1]$$

With c a constant describing the relation between robot speed and image motion. On Figure 3, this image motion field is shown as calculated by the right robot camera.

To come to one single numerical value m representing the amount of motion in a camera image, the resulting motion field is summed over the whole image domain, giving $m = \sum_{k=1}^{\text{all pixels}} m_k$. This calculation is performed once for the left camera and once for the right camera image, leading to two distinct image motion estimates.

a) b) c)

Figure 3. *Image motion field of right camera: a) image at t_{i-1}; b) image at t_i; c) image motion field*

3. Robot control architecture

The control architecture describes the strategy to combine the three main capabilities of an intelligent mobile agent: sensing, reasoning and actuation. These three capabilities have to be integrated in a coherent framework in order for the mobile agent to perform a certain task adequately. A number of control strategies can be set up, varying from simple serial sense-model-plan-act strategies to complex hybrid methods. As the robot task involves achieving goals which are not defined at designtime, a purely reactive strategy will not suffice in our case, unless it is extremely (and thus prohibitively) complex. A robot which is required to navigate to various locations specified by the user at runtime cannot perform this task reactively unless its state-action policy includes states for each of the possible goal locations. Purely reactive approaches achieve great efficiency because of their minimal computation. However, since they have so little representational power, they lack runtime flexibility. To combine the advantages of purely reactive and planner-based approaches, this research work aims at implementing a hybrid control strategy which fuses a behaviour-based controller for autonomous navigation with automation aspects for mine scanning. A behaviour-based robot control architecture was chosen because of the flexible and modular nature of behaviour-based controllers, facilitating the design process. Additionally, a common property of behaviorbased systems is their distributed nature; they consist of a collection of parallel, concurrently executing behaviors devoid of a centralized arbiter or reasoner. The performance of the behaviour-based controller depends on the implementation of the individual behaviours as well as on the method chosen to solve the behaviour fusion or action selection problem.

3.1. *General architecture*

The working principle of the proposed control architecture is sketched on Figure 4. There are three distinctive modules: Navigation (on the right side on Figure 4), Mine Detection - Scanning (in the middle on Figure 4) and Metal Detection (on the left side on Figure 4). These three processes are controlled by a watchdog, the robot motion scheduler, which manages the execution of each module and decides on the commands to be sent to the robot actuators. This robot motion scheduler is explained more in detail in Section 3.5. In normal situations (when no suspicious objects are detected), the leftmost and rightmost modules (Metal Detection and Navigation) are both active. In this situation, the metal detector senses continuously for metal in the soil and the navigation module steers the robot towards a certain goal, while avoiding obstacles. When metal is found, the central module, Mine Detection, is activated by the robot motion scheduler. In this case, the robot is stopped as the navigation module is no longer active and the cartesian scanner searches for metal in a predefined pattern. The output of this scanning procedure is used by mine imaging tools to determine whether the suspected metal object is a mine or not. This information is used by the navigation procedure as extra sensor input, as mines need to be avoided too.

In the following sections, the different groups depicted in Figure 4 are presented more in detail.

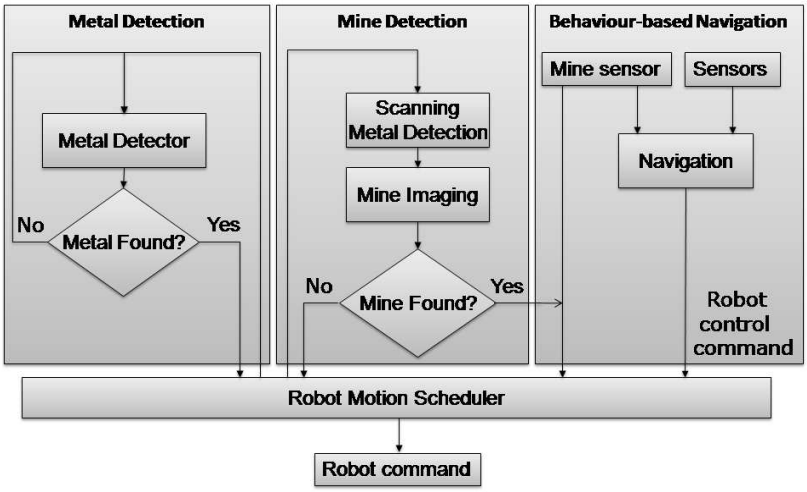

Figure 4. *The general robot control architecture*

3.2. *Metal detection*

The metal detector scans for metal in the soil. If no metal is found, it keeps on doing this and the robot keeps on moving. If a metal is found, this is reported to the robot motion scheduler, which takes the appropriative actions. We will not elaborate here on the implementation of a metal detecting sensor, as it is out of the scope of this paper.

3.3. *Mine detection*

The Cartesian scanning mechanism makes a 2D scan with the metal detector. Combining all the individual metal detector measurements, on obtains an image of the soil. An example of such an image is given in Figure 5. In this image, the bright oval areas on Figure 5 have a mine-like shape and size, so they need to be treated as suspected objects. Such an analysis is performed automatically by mine imaging tools, which determine the likelihood of mine occurrence and the exact position of eventual mines. It is obvious that the diverse nature of modern mines can make this process very difficult. The Royal Military Academy, leading the Belgian HUDEM project, has been focussing for several years on the development of new data processing and fu-

sion algorithms for mine detection (Milisavljevic, 1999), on the improvement of mine detecting sensors and on robotic systems that carry these sensors. This paper focusses on the third research topic, so we will not discuss these mine detection algorithms any further.

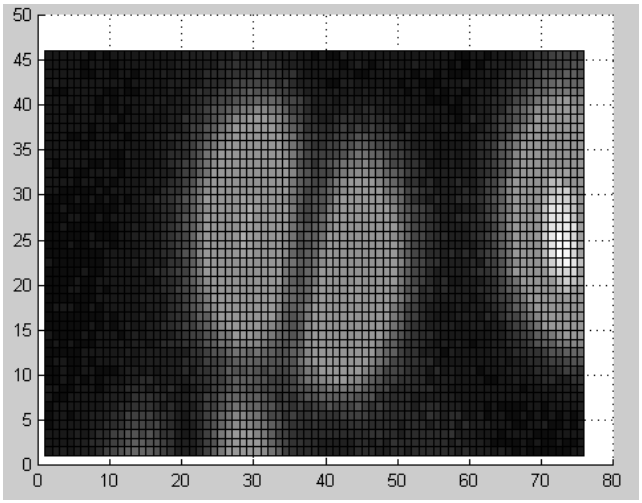

Figure 5. *Example result of an image of the soil obtained by making a cartesian scan with a metal detector*

If a mine is found, this is reported to the robot motion scheduler, which takes the appropriative actions. In addition to this the Mine detector acts as a sensor for mines, as it returns the locations of mines, which need to be considered as obstacles themselves.

3.4. *Behaviour-based navigation controller*

For the Robudem robot, three main sensing capabilities are installed: odometry, stereo vision and image motion analysis. These three sensing capabilities are processed by separate behaviours. In this context, the odometry information is used for guiding the robot to a given goal position, while the visual sensors are used for obstacle avoidance.The main advantage of using behaviour-based approaches is that each of the different behaviours can be designed and implemented separately. The general scheme of the behaviour-based navigation module is shown on Figure 6. As indicated by Figure 6, the behaviour-based controller has a hierarchical structure. At the lowest level, we can find the sensor input data for the behaviour-based controller consisting of odometry information and stereo images. From the visual input stream, two abstract

sensors are deduced as explained in Section 2: stereo vision and image motion analysis. This leads to three main behaviours defined for robot navigation: *goal seeking*, *avoiding obstacles using stereo vision* data and *avoiding obstacles using image motion analysis*. As the architecture is modular, more behaviours can be added efficiently. Behaviour fusion then extracts consistent control commands from these individual behaviours. The output of this behaviour-based navigation planner are velocity (V) and turn angle (T_{Front}, T_{Rear}) commands. The reason for having two different turn angle commands is related to the mechanical structure of the Robudem robot, which has a two-by-two differential drive system, meaning front and back wheels can be given a different turning angle, allowing for highly flexible maneuvering on difficult terrain. This also allows for different drive modes to be adopted by the robot. It is our final aim to let the behavioural controller decide which is the best drive mode given the terrain circumstances. For now, the user can select on the graphical user interface the desired drive mode and T_{Front} and T_{Rear} are calculated accordingly.

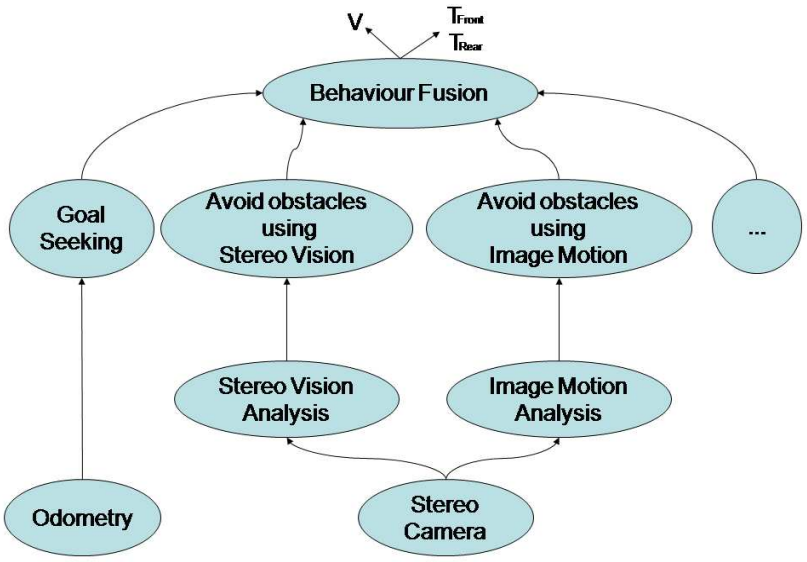

Figure 6. *General scheme of the behaviour-based navigation module*

We will now explain the different components of the behaviour-based controller depicted in Figure 6 more in detail in a bottom up approach, so starting with explaining the different individual behaviours. Each of these behaviours calculates its own velocity and turn angle for the robot. These velocity and turn prescripts are calculated at behaviour level as follows:

– the *avoid obstacles using stereo vision* behaviour receives as data d_l and d_r, the distances to obstacles on the left and right side of the robot, by stereo vision analysis. The smaller the distance to obstacles, the more carefully and slowly the robot must move. The velocity V_S is therefore directly proportional to the mean of the measured distances left and right as expressed by Equation [2]:

$$V_S = c_S \frac{d_l + d_r}{2} \quad [2]$$

with c_S a normalization constant. When the distance to obstacles on the left side is larger than the distance to obstacles on the right side, the robot should avoid these obstacles on the right by turning to the left. This relation between robot turn angle and distances to obstacles is expressed by Equation [3]:

$$T_S = c_S \left(d_l - d_r\right) \quad [3]$$

as such, the distance to obstacles is maximized;

– the *avoid obstacles using image motion* behaviour receives as data m_l and m_r, the movement in the left and the right camera measured by the image motion analysis. The more movement in the image, the more probable there are objects close to the robot, so the velocity should be lowered. The robot speed V_M is as such inversely proportional to the image motion as expressed by Equation [4]:

$$V_M = 1 - c_M \frac{m_l + m_r}{2} \quad [4]$$

with c_M a normalization constant. When the movement on the left side is larger than the movement on the right side, the robot should avoid these probable obstacles on the left by turning to the right, meaning the robot turn angle can be calculated from the movement in the images as given by Equation [5]:

$$T_M = 1 - c_M \left(m_l - m_r\right) \quad [5]$$

– the goal seeking behaviour receives as data the estimate of the robot position (x_R, y_R) and orientation θ_R as calculated by the odometry and robot kinematics. The estimation of the robot motion and posture is done via a non-linear method named the *dynamic extension algorithm*, as presented in (Habumuremyi *et al.*, 2005). The robot position is compared with the desired goal position (x_G, y_G) and orientation θ_G and a velocity V_P and turn angle T_P command are calculated from this information. The velocity of the robot should be maximal if the robot is far enough from the goal and should be reduced when approaching the goal. Therefore, the calculation of the velocity command is split in two parts:

$$V_P = \begin{cases} v_{\max} & \text{if} \quad d_{\text{target}} > d_{\text{threshold}} \\ \frac{d_{\text{target}}}{d_{\text{threshold}}} v_{\max} & \text{if} \quad d_{\text{target}} < d_{\text{threshold}} \end{cases} \quad [6]$$

with d_{target} the distance to the goal, calculated from the robot and goal position and $d_{\text{threshold}}$ a certain security distance above which the robot is allowed to drive with the maximum robot velocity $v_{\max}$. The turn angle command can be calculated by expressing that the heading of the robot and the target should be the same:

$$T_P = -\left(\frac{\theta_R - \theta_G}{c_P}\right)^3 \quad [7]$$

with c_P a normalization constant.

Behaviour fusion deals with distilling a coherent robot command from the individual behaviour prescripts. Therefore, the outputs of the behaviours are weighed according to the activity level A of the specific behaviour as expressed by Equations [8] and [9]. The activity level of a behaviour describes to which degree this behaviour is relevant for the calculation of the final robot command.

$$V = (1 - A_E)\left(\frac{A_S V_S + A_M V_M + A_P V_P}{A_S + A_M + A_P}\right) \quad [8]$$

$$T = A_{F,R}\left(\frac{A_S T_S + A_M T_M + A_P T_P}{A_S + A_M + A_P}\right) \quad [9]$$

with:

– V and T respectively the velocity and turning command for the robot;

– A_E the activity level for the *Emergency Stop* which allows the user to de-activate the robot at all time-instances via the user interface for security reasons;

– A_F, A_R the activity levels for *Front Steering* and *Rear Steering*, which decide on the drive mode which will be adopted by the robot;

– A_S, A_M, A_P the activity levels for, respectively, the *Obstacle Avoidance using Stereo Vision*, *Obstacle Avoidance using Image Motion Analysis* and Goal Seeking behaviour;

– V_S, V_M, V_P the Velocity commands from, respectively, the *Obstacle Avoidance using Stereo Vision*, *Obstacle Avoidance using Image Motion Analysis* and Goal Seeking behaviour;

– T_S, T_M and T_P the turn angle commands coming from, respectively, the *Obstacle Avoidance using Stereo Vision*, *Obstacle Avoidance using Image Motion Analysis* and Goal Seeking behaviour.

A major issue in the design of behaviour-based control systems is the formulation of effective mechanisms for coordination of the behaviours' activities into strategies for rational and coherent behavior. Such fusion mechanisms are based upon the calculation of the activity levels as presented above. These activity levels should reflect the relevance of the specific behaviour. The principle behind the calculation of the activity levels is that the output of a behaviour should be stable over time in order to trust it. Therefore, the degree of relevance or activity is calculated by observing the history of the output - a velocity and turn angle - of each behaviour. This history-analysis is performed by comparing the current output to a running average of previous out-

puts, which is transformed to a normalized standard deviation. For the stereo vision behaviour, these standard deviations are expressed by Equations [10] and [11]:

$$\sigma_{S,V} = c_V \sum_{k=i-h}^{i} \left(V_{S,k} - \frac{\sum_{j=1}^{N} V_{S,j}}{N} \right)^2 \quad [10]$$

$$\sigma_{S,T} = c_T \sum_{k=i-h}^{i} \left(T_{S,k} - \frac{\sum_{j=1}^{N} T_{S,j}}{N} \right)^2 \quad [11]$$

with c_V and c_T two normalization constants.

The larger this standard deviation, the more unstable the output values of the behaviour are, so the less they can be trusted. The same approach is followed for the image motion (subscript M) and the goal seeking (subscript P) behaviours. This leads to an estimate for the activity levels as expressed by Equation [12]:

$$A_{S/M/P} = \left(1 - \sigma_{S/M/P,V}\right) + \left(1 - \sigma_{S/M/P,T}\right) \quad [12]$$

For stability reasons, the activity level is initialized at a certain value (in general 0.5) and this estimate is then iteratively improved. The presented approach towards calculating the activity levels has some major advantages over traditional methods. In comparison to classic fuzzy logic data fusion approaches where the number of rules grows in general rapidly with the number of inputs, the complexity only increases linearly in this approach. Moreover, it is very modular, easily updateable and the result can be calculated very quickly. One of the disadvantages of this method is that the normalization constants c_i must be user-determined. However, this process of parameter tuning only needs to be performed once for each abstract sensor, as the normalization constants c_i are independent of one another.

As such, all information necessary to calculate Equations [8] and [9] is known, meaning that the behaviour-based navigation controller can propose a robot control command in the form of a velocity and turning command. This data is given as an input to the robot motion scheduler which will execute it, unless another module has a higher priority task (and trajectory) to perform.

3.5. *The robot motion scheduler*

The robot motion scheduler needs to decide which of the modules is executed and which of them can influence the robot actuators through robot commands. Its flow diagram is sketched on Figure 7.

There are two main paths through the robot scheduler, one for the (normal) situation of exploring while avoiding obstacles and detecting metals and one for the situation where metal is found and more thorough investigation is needed (mine detection) while the robot is standing still.

In a normal situation, occurring *e.g.* in an initial situation (default inputs), the metal detector is activated but the scanner is turned off. The navigation module returns at all time instances a robot action prescript, as this module loops infinitely without interaction with the other modules. This trajectory is set as the trajectory to be executed, but with a low priority.

If the *metal found* trigger is given, the metal detector is switched off. The trajectory for the robot is set to a predefined movement, more specifically, to back off a little. This is done to be able to centre the scanning metal detection better around the suspicious object. This trajectory has a high priority. When this movement is completed, the robot is halted, by giving a "no movement" trajectory with a high priority. Finally, the scanning metal detection module is activated.

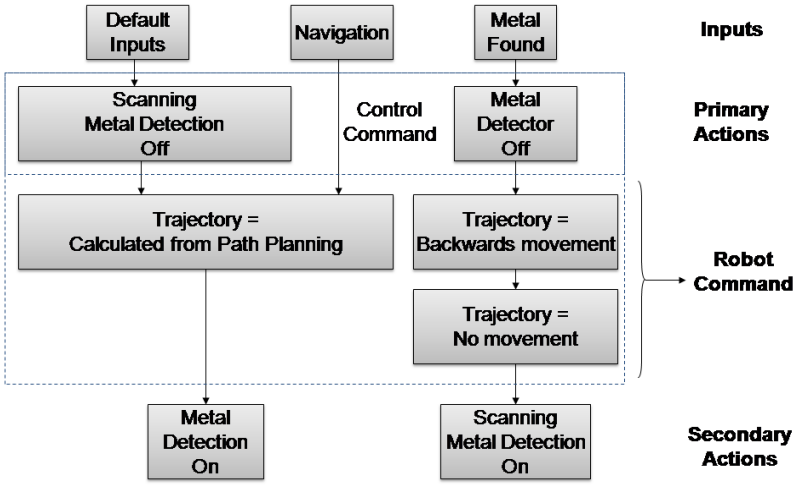

Figure 7. *The control architecture for the robot motion scheduler*

4. A modular software architecture

4.1. *An introduction to MCA*

As control architectures which aim to mimic human thinking tend to rapidly become highly complex, the use of a flexible, extendable and real-time capable software architecture is very important. Such a software architecture has to ease the development and reuse of software components. The chosen software architecture, MCA (Modular Controller Architecture) as presented by Scholl in (Scholl *et al.*, 2002), achieves this by employing simple modules with standardized interfaces. The communication in MCA is managed *via* transporting edges that connect modules together. The main programs only consist of constructing modules that are connected via edges and pooled into a group. This results in an equal programming on all system levels. As modules can be integrated both on Windows, Linux and on RT-Linux without changes, they can be developed on Linux-side and then transferred later to RT-Linux. As errors in RT-Linux lead to system-hangs this development strategy prevents from many reboot cycles and results in faster software development.

Each MCA module has a structure as shown on Figure 8 and is determined by four connectors with the outside world: Sensor input (left below), Sensor output (left top), Control Input (right top), Control Output (right below). As a result sensor data streams up, control commands stream down. The Sensor input and output are connected through a Sense procedure which enables to process the sensor data and the Control input and output are connected through a Control procedure which enables to process the control commands. On Figure 8, sensor data flow is shown in yellow, control command flow in red.

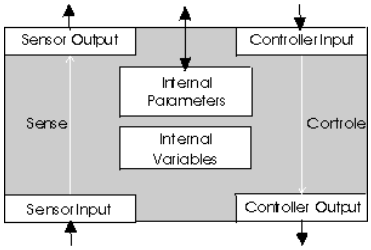

Figure 8. *An MCA module made up of a Sensor Input and Sensor Output edge, linked by a Sense procedure and a Control Input and Control Output edge, linked by a Control procedure*

This modular structure is particularly convenient for behaviour-based architectures as the individual behaviours translate easily to corresponding MCA-modules. This is why we chose MCA above other software architectures.

4.2. *The general control architecture*

The general robot control architecture, as presented in Section 3.1, is translated into an MCA scheme as shown on Figure 9. The architecture features a layered hierarchical structure with three levels. The first level consists of low level sensor processing modules such as the stereo framegrabber and the metal detector, together with low-level actuator controlling modules such as the uplink to the Robudem robot and to the cartesian scanner. One level higher in the hierarchy, high-level abstract sensors such as the stereo vision module, the image motion analysis module and the mine imaging module process the data received by the low-level sensors. These image processing steps were explained more in detail in Section 2. Also the robot kinematics and dynamics module which provides the odometry data can be found at this level. It can be observed that when changing the robot platform, only the robot kinematics and dynamics module and the robot uplink need to be rewritten. This allows for fast and easy development of control architectures for multiple robotic platforms, based upon a common codebase. Eventually, the high-level abstract sensors output their data to the navigation controller, which decides on the robot actions, in conjunction with the robot motion scheduler, as explained in Section 3.1. The structure of this navigation controller in MCA is explained in Section 4.3.

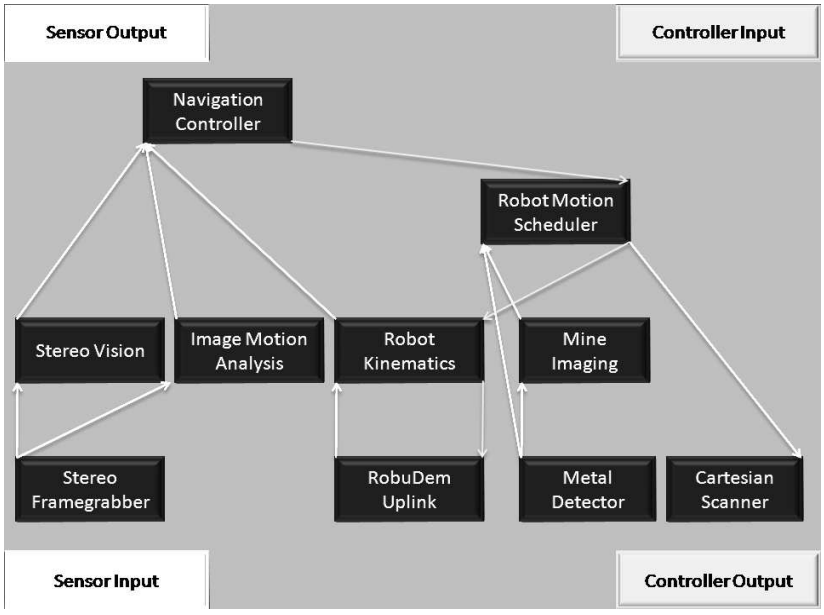

Figure 9. *General MCA scheme for the robot controller*

4.3. *The general behaviour-based navigation controller*

The MCA scheme for the behaviour-based controller is shown on Figure 10. It consists of the three main behaviours explained before, controlled (fused) by a *BehaviourSelector* module.

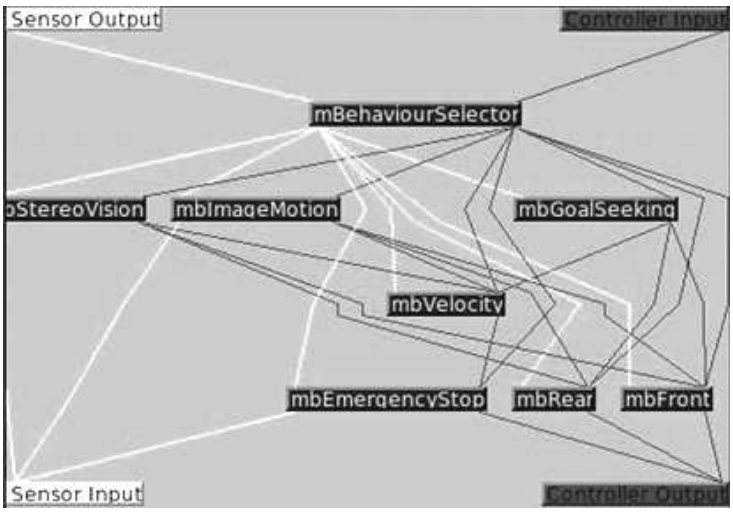

Figure 10. *MCA scheme for the behaviour-based controller*

The sensory input received by the behaviour-based controller and represented on the bottom left of Figure 10, consists of the distances to obstacles: d_l, d_c, d_r from the *stereo vision* module; the motion in the left and right camera m_l, m_r from the *image motion analysis* module and the robot position (x_R, y_R) and orientation θ_R as estimated by the *odometry*. This data is processed by the *stereo vision*, *image motion* and *goal seeking* behaviours, outputting a velocity and turning prescript. The *BehaviourSelector* module receives as input the output of the different behaviours and calculates the activity levels for each of these behaviours according to Equations [12].

All of the three main behaviours also send the calculated velocity setup value to the *Velocity* module, where the data fusion occurs, based on the activity levels which were calculated by the *BehaviourSelector* module. The *EmergencyStop* behaviour, which can be triggered by the user or when an obstacle is detected within a security distance, ensures that the velocity command is only transferred to the robot in safe conditions.

For the turning behaviour, a similar approach is followed; only in this case there are two separate fusion behaviours the navigation behaviours can send their results to. As explained before, this is the case because the robot can assume different drive modes. For example, for *car-like* driving, only the front wheels are steered and for

crab-like driving, all wheels turn in the same direction. It is the *BehaviourSelector* which decides on the drive mode based on user input and sets the activity levels for front and rear steering accordingly.

5. Results

The result of the presented research project is a generic robot control architecture which can be used for the control of different real and simulated robots. Of course, when switching robots, some modules need to be interchanged, such as for example the *RobuDem Uplink* module, which is responsible for translating MCA commands to robot commands and back. The global achievements of this research project can be discussed by taking a look at the Graphical User Interface (GUI) in Figure 11, as it shows the important features of the Robudem controller.

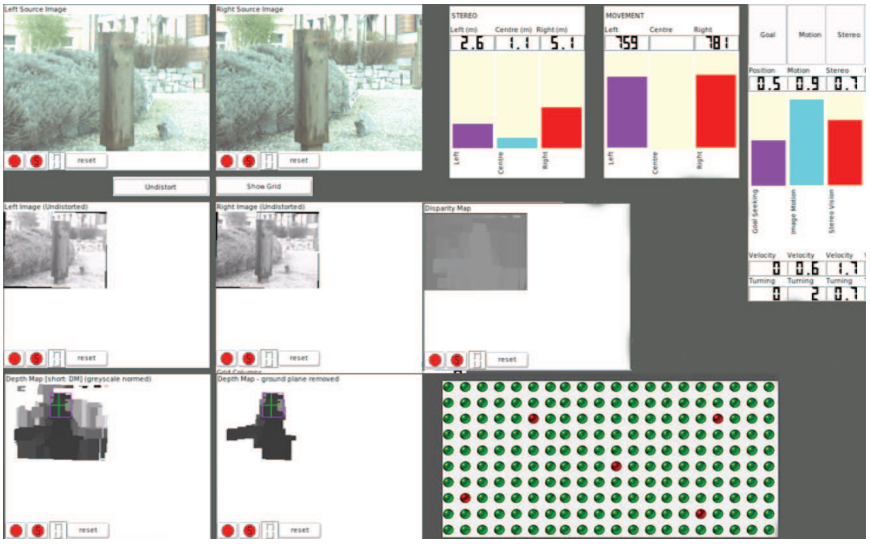

Figure 11. *User Interface of the Robudem navigation controller*

On the upper left, the stereo camera images are visible, which are rectified to facilitate the calculation of the depth map. The dense depth map is then postprocessed, as is shown on the images on the lower left. In the top middle of the interface, the measurements of the abstract visual sensors stereo vision and image motion analysis are shown using colored bars. These indicate for the stereo vision sensor the distances to obstacles on the left, middle and centre and for the image motion sensor the motion in the left and right camera image.

At the right of the interface, the activity levels of the different behaviours are shown. As can be noticed, the BehaviourSelector has found here that the Image-Motion behaviour delivers more reliable results than the StereoVision behaviour. This is in this case due to the lack of texture in the camera images, which renders the dense depth map estimation less robust. On the lower right of this robot control interface the map of suspected mine locations is shown, as detected by the robot metal/mine detector. Red dots indicate suspected mine areas.

6. Conclusions

In this paper, we presented three main aspects of the design process of an intelligent autonomous robotic agent for demining interventions: visual sensing, behaviour-based control and the software architecture. Multiple visual cues, stereo vision and image motion analysis, were integrated into the robot control and software architecture. A behaviour based control architecture was proposed, using statistical reasoning to solve the action selection problem. All components were implemented using a modular software architecture to achieve a future-proof design. The integration of these aspects enables the robot to search for mines in a designated area while avoiding obstacles.

The purely reactive nature of the presented framework is certainly the major drawback of the approach. Therefore, it is the aim of future research to combine the presented behavior-based approach with model based reasoning. For this, we are developing a visual simultaneous localization and mapping (V-SLAM) system, which can automatically map the environment and place the robot in this map. Moreover, a differential GPS system and an orientation sensor will be placed on the robot. By integrating the information from these sensors with visual data in a SLAM context, the precision of the robot localisation will be further increased.

7. References

Arkin R., " Motor Schema based navigation for a mobile robot: An approach to programming by behavior", *IEEE International Conference on Robotics and Automation*, p. 264-271, 1987.

Birchfield S., Klt: An implementation of the kanade-lucas-tomasi feature tracker, Technical report, http://www.ces.clemson.edu/stb/klt/, Jan. , 1997.

Brooks R., " A robust layered control system for a mobile robot", *IEEE Journal of Robotics and Automation*, vol. 2, n° 1, p. 14-23, 1986.

Bruyninckx H., " Open robot control software: the OROCOS project", *Proc. Int. Conf. on Robotics and Automation*, vol. 3, p. 2523-2528, 2001.

Colon E., Sahli H., Baudoin Y., " CoRoBa, a multi mobile robot control and simu-lation framework", *International Journal of Advanced Robotic Systems*, vol. 3, n° 1, p. 73-78, 2006.

Davison A., Reid I., " MonoSLAM: Real-Time Single Camera SLAM", *IEEE Trans. on Pattern analysis and machine intelligence*, vol. 29, n° 6, p. to appear, 2007.

DeCubber G., Berrabah S., Sahli H., " Color-Based Visual Servoing Under Varying Illumination Conditions", *Robotics and Autonomous Systems*, vol. 47, n° 3, p. 225-249, 2004.

DeSouza G., Kak A., " Vision for Mobile Robot Navigation: A Survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, n° 2, p. 237-267, 2002.

Doroftei D., Behavior based Robot Navigation Techniques - State of the Art, Technical report and lecture notes, Royal Military Academy of Belgium, Oct. , 2006.

Enescu V., DeCubber G., Cauwerts K., Berrabah S., Sahli H., Nuttin M., " Active Stereo Vision-based Mobile Robot Navigation for Person Tracking", *Proc. Int. Conf. on Informatics in Control, Automation and Robotics*, vol. 2, p. 32-39, 2005.

Gerkey B., Vaughan R., Howard A., " The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems", *Proc. Int. Conf. on Advanced Robotics*, Portugal, 2003.

Habumuremyi J., Houpin J., " Model of a wheeled robot named Robudem and design of a state feedback controller for its posture tracking: simulation and experiment", *ISMCR05*, Brussels, Belgium, November, 2005.

Hong P., Sahli H., Colon E., Baudoin Y., " Visual Servoing for Robot Navigation", *Proc. 3rd Int. Conf. on Climbing and Walking Robots*, Germany, p. 255-264, 2001.

Maes P., How To Do The Right Thing, Technical report ne 43-836, AI-Laboratory, Massachusetts Institute of Technology, 1989.

Mataric M., " Behavior-based control: Examples from navigation, learning and group behaviour", *Journal of Experimental en Theoretical Artificial Intelligence, special issue on Software Architectures for Physical Agents*, vol. 9, n° 2-3, p. 323-336, 1997.

Medeiros A., " A survey of control architectures for autonomous mobile robots", *Journal of the Brazilian Computer Society*, 1998.

Milisavljevic N., " Mine shape detection and data fusion considerations", *Proc. of the HUDEM Workshop*, Brussels, Belgium, 1999.

Park C., Kim S., Paik J., " Stereo vision-based autonomous mobile robot", *Intelligent Robots and Computer Vision XXIII: Algorithms, Techniques, and Active Vision*, vol. 6006, p. 256-264, 2005.

Piccardi M., " Background subtraction techniques: a review", *Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, vol. 4, p. 3099-3104, 2004.

Pirjanian P., Behavior coordination mechanisms - state-of-the-art, Technical report and Lecture Notes n° Tech. Report IRIS-99-375, Institute of Robotics and Intelligent Systems, University of Southern California, Oct. , 1999.

Pirjanian P., Huntsberger T., Trebi-Ollennu A., Aghazarian H., Das H., Joshi S., Schenker P., " CAMPOUT: a control architecture for multirobot planetary outposts", *Proc. SPIE Conf. Sensor Fusion and Decentralized Control in Robotic Systems*, USA, 2000.

Schafer H., Proetzsch M., Berns K., " Stereo-Vision-Based Obstacle Avoidance in Rough Outdoor Terrain", *International Symposium on Motor Control and Robotics*, 2005.

Schlegel C., Illmann J., Jaberg H., Schuster H., Worz R., " Integrating vision-based behaviors with an autonomous robot", *Videre*, 2000.

Scholl K., Gassmann B., Albiez J., Zollner. J., " MCA - Modular Controller Architecture", *Robotik 2002*, 2002.

ANNEXE POUR LE SERVICE FABRICATION
A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :

JESA - 43/2009. Interactions homme-machine

2. AUTEURS :

Daniela Doroftei[*, **] — *Eric Colon*[*] — *Yvan Baudoin*[*] — *Hichem Sahli*[**]

3. TITRE DE L'ARTICLE :

Development of a behaviour-based control and software architecture for a visually guided mine detection robot

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :

Towards mine detection robots

5. DATE DE CETTE VERSION :

February 23, 2009

6. COORDONNÉES DES AUTEURS :

– adresse postale :

[*] Royal Military Academy, Department of Mechanical Engineering (MSTA)
Avenue de la Renaissance 30, B1000 Brussels, Belgium
{daniela.doroftei;eric.colon;yvan.baudoin}@rma.ac.be

[**] Vrije Universiteit Brussel, Department of Electronics and Informatics (ETRO)
Pleinlaan 2, B1040 Brussels, Belgium
hsahli@etro.vub.ac.be

– téléphone : 00 322 742 65 54

– télécopie : 00 322 742 65 47

– e-mail : -

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :

LaTeX, avec le fichier de style `article-hermes.cls`,
version 1.23 du 17/11/2005.

8. FORMULAIRE DE COPYRIGHT :

Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :
`http://www.revuesonline.com`