Smart, Simple and Low-Cost Control of planetary exploration rovers

Robin G.J. Biesbroek and Arne Y.J. Matthyssen

JAQAR Space Engineering
Hector Berliozkade 68, 2551 XR Den Haag, The Netherlands
Telephone: +31.70.3978665, Fax: +31.70.3978665
General Information: info@jaqar.demon.nl
http://www.jaqar.demon.nl

1. ABSTRACT

This paper focuses on a control architecture for (micro)-rovers that can be used for planetary or small body exploration; an architecture of which the design is entirely based on the objective of creating a system that is robust, simple and cheap. This robustness and simplicity is achieved by effectively managing and allocating the control tasks to either the on-ground part or the on-board part of the system. Important features of this system are the level of rover autonomy, the use of classifier systems to generate commands (based on the knowledge of some basic rules, the rover and its environment), a contingency list. The system is foreseen to be implemented on a PC using off the shelf software.
The concept incorporates human decision making and control together with a learning system and a rover with autonomous decision making capabilities.

2. INTRODUCTION

In the attempt to provide more human-like intelligence to the exploration rovers the stress most of the time lies on the enhancement of the "tele-presence" through virtual reality, stereo vision, sound, smell sensors etc. This paper presents a concept that gives the possibility to the system to learn from the decision of the operator and the interaction with the environment and change its decisions if necessary, as a human would do.
The concept was conceived with the knowledge of the reserved attitude of the planetary exploration society, regarding allowing too much artificial intelligence in expensive and complex missions. It is our feeling that the proposed concept does not allow any uncontrollable intelligence in the system that would block out human authority and provide additional risks for the mission .

3. CONTROL CONCEPT

The control concept is based on effectively managing and allocating the control tasks to either the ground segment or the rover segment of the system, together with the use of classifier systems [RD 1] to generate commands and incorporate the ability to learn.
Two different types of control are defined:

- human interactive
- autonomous control.

During human interactive control the operator commands the rover which executes the command as good as possible (although it was not the scope of this work, the human interaction control mode can be an interactive autonomy mode, in which high level commands are send and the rover has some kind of autonomy, e.g. based on behaviours, to execute the command). This control-type is the normal operations mode and allows a man-in-the-loop at all times. The learning system based on classifier systems can be toggled on or off.

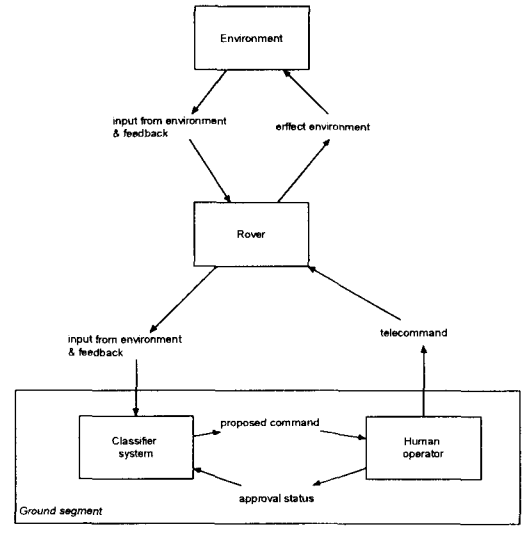

Figure 3.1: Control concept.

In the autonomous mode of operation the system is actually reduced to only a rover part. This phase only occurs when the human control has stopped e.g. due to unexpected loss of contact (LOC), in this case the rover becomes autonomous and uses the contingency list to diagnose the reason for the contingency, and take the appropriate actions to secure survival.
Note that autonomous control does not start immediately after LOC, but after a period of time (for example 1 hour); this allows the user or control centre a period of time to try and re-establish contact. This is incorporated using a trigger function built in the rover. Learning or not during autonomous operations is an issue that is briefly addressed in §5.

To provide the system with the proposed control several items are needed:

1. Simple commands
2. Classifier systems
3. Contingency list
4. Environment maps

Proc. Fifth International Symposium on Artificial Intelligence, Robotics and Automation in Space, 1-3 June 1999 (ESA SP-440)

3.1 Simple commands

The use of simple, parameter-less commands allow the use of classifier systems (see §3) to be implemented in the system. When stored in binary form, a set of simple TC's for the 4 legged PROLERO micro-rover [RD2] could look like:

Move front left leg	
Forward:	00000001
Backward:	00000010
Move front right leg	
Forward:	00000100
Backward:	00001000
Move back left leg	
Forward:	00010000
Backward:	00100000
Move back right leg	
Forward:	01000000
Backward:	10000000

These commands are of course not the commands one wants to sent to the rover. One of the objectives of the work was to evaluate the possibility to use classifier systems to generate based on these "leg" commands, more complex "steering" commands. For example:

Normal forward:	01010101 001
Normal backward	10101010 001
Climbing forward:	01010101 010
Climbing backward	10101010 010
Descending forward:	01010101 100
Descending backward	10101010 100
Turn left:	01100110 000
Turn right	10011001 000

as well as commands like "Turn over"; "Climb step"; "Take monitoring position" etc.

To get to the "steering" commands the system has to go through a learning phase, using the response of the operator to the proposed command (combination of the 8 leg commands), in order to come to a "steering" command that is appropriate in the current situation (defined by the telemetry (TM)).
If the system has enough time, the system allows , based on its "leg" & "steering" commands, the generation of new walking methods (see §5).

During non-learning mode the system will, depending on the telemetry and knowledge of the environment, propose one of these commands to the operator.

Apart from simple TC's, the classifier system requires input from the environment (telemetry (TM)) as simple string rules. If the sensor output can have many different values, the range will be divided into different classes e.g. low, medium, high in order to have a limited number of messages, that can be binary represented (see §3.4) .

3.2 Classifier systems

[RD1]
Classifier systems give the system the possibility to propose commands and learn from the responds of the operator and the TM. A classifier system (CS) is a machine learning system that learns syntactically simple string rules, called classifiers which guide the system's performance in an arbitrary environment. The CS has developed out of the merging of expert systems and genetic algorithms.
Figure 3.1 shows the components of a CS and its interaction with the environment. The CS receives information about the environment, performs internal processing and then effects the environment. In learning mode, the CS uses feedback about the effect on the environment to learn from the experience. If no feedback is provided, the CS is in application mode (non-learning).

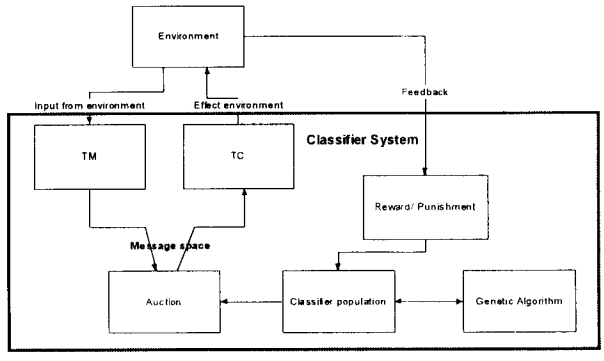

Figure 3.2: Classifier system components and interaction with the environment.

A classifier system has 3 major components:
1. Rule & message sub-system.
2. Apportionment of credit sub-system.
3. Classifier discovery mechanisms.

Detailed information on how a CS is used for rover control is given in chapter 4.

3.3 Contingency list

A contingency is a rover life threatening situation, e.g. sudden LOC, power too low, temperature too high/low etc. This means rover specific "housekeeping" parameters define the contingency. In this work the fact that e.g. the rover is turned upside down is not considered as a contingency, the system can react with its normal classifiers to this situation, or learn to get out. The contingency list is only used to detect when to go into autonomous mode. The list consists of several conditions and the appropriate action to be taken, needed for the rover to survive, in case of such a contingency condition.

3.4 Environment maps

To allow the system to propose a command which is not out of the blue, some knowledge of the environment which the rover will encounter has to be available a priori. To stay within the goal of the concept, (robust,

simple & cheap) no fancy environment mapping techniques can be used. The information about the environment can e.g. be represented as fuzzy maps [RD3]. Fuzzy map representation incorporates and allows handling of the lack of information about the environment and the in-accuracy that comes with it, and such maps do not require much disk-space to store.

The environment representation can be seen as follows. A map, divided in cells, consisting of useful parameters, e.g. temperature, height, slope, soil type, danger etc., can exist. The parameters will not consist of values, but binary re-presentable values like e.g.:

00000001 Danger zone
00000010 Hard ground
00000100 Medium-hard ground
00001000 Soft ground
00010000 Steep slope
00100000 Link status OK
01000000 Temperature low
10000000 Power low
etc.

Three types of maps can be distinguished "terrain maps", "housekeeping maps" and "direction map". Information like danger-zones, slopes, soil characteristics etc. are defined before operations, it can be envisaged that the rover sensors provide enough information during operations to update these "terrain maps". "Housekeeping maps" are generated, during operations, on-ground using the rover TM (link status, power, internal temperature etc.), and regularly uploaded to the rover. This to update or replace its current maps, to insure that the rover in case of contingency has the most up to date maps, to find a safe place.

The "direction map" consists of the direction from each cell to the target position. This map can be generated manually by the human operator or force field can be envisaged.

In creating a command the system looks at the TM to be expected (included in the maps) for the next surrounding cells; going if possible to the next cell with the direction (direction map) pointing to the mission target.

4. GROUND CONTROL

The functionalities of the ground system are threefold:

- Creation and uploading of TC's
- Storage of the rule-message system
- Receiving of TM

The creation of the TC's is performed using a CS as described in the previous section. The TM is received in a simple string-format such as the example given in section 3.1, and is regarded as the 'message'. The CS seeks the classifiers that match the message, and holds an auction: all the matching classifiers submit a bid, and the classifier with the highest bid is proposed as a TC to the user.

1. Rule & message sub-system.

Each classifier consists of a rule in the form of:

IF (<condition1>&<condition2>&...&<conditionN>) THEN (<action>)

where

<condition> is encoded as a string from the alphabet {0, 1, #}
<action> is encoded as a string from the alphabet {0, 1} and forms a TC.

The '#' symbol acts as a 'don't care' in the condition, matching either a 0 or 1, and allows for more general rules.
Each rule has an associated strength giving measure to the rule's past performance in the environment in which it is learning.
The messages are generated from the environment: the TM in a simple string-format, and match the condition part of the classifier rule.
For example, the following classifier could exist (with ':' denoting the break between the conditions and the action):

1#####01:10101010 001

which would, using the example codes of the previous section mean: if there is an obstacle in front of the rover (danger zone), and the rover is low on power, but the rover is not on a hard ground, then walk backwards normally . The classifier doesn't care what inclination the rover has, or if the rover is on medium-hard ground or soft ground.

2. Apportionment of credit sub-system.

This sub-system deals with the modifications in strength of classifiers as the CS learns. These modifications occur via three mechanisms:

- Auction
- Reinforcement & punishment
- Taxation

When the CS receives messages from the environment, all the classifiers which match one or more of the messages compete, by submitting a 'bid' in an 'auction' to determine a victorious classifier that will effect the environment i.e. propose its TC to the user. The bid is a function of the classifier's strength and specificity (number of non-'#' symbols). Only the bid of the victorious classifier is paid, and therefore the victorious classifier has its strength decreased by the amount of its winning bid.
The user will then approve or reject the proposed TC. If approved, the TC will be send to the rover. If the CS is

in 'learning' mode (allow feedback), the strength of the victorious classifier is increased when the TC is approved, or decreased when the TC is rejected. If the TC is rejected, the auction could be repeated until the user approves one. This has the advantage for the learning system to learn faster, since it receives more feedback, however it may take many proposals until the user approves a TC, if the TC the user has in mind is not obvious to the system. In that case, the user could choose the TC by hand. The CS could look up a matching classifier and regard it as a bid from that classifier, therefore increasing its strength.
Taxation occurs to prevent the classifiers from being cluttered with artificial high strength classifiers of little or no utility. Taxation is levied on each classifier per iteration (life tax) and on each classifier that submits a bid during an auction (bid tax).

3. Classifier discovery mechanisms.
The classifier discovery mechanisms consists of a genetic algorithm and a triggered cover detector operator.
A Genetic Algorithm (GA) [RD4] is an optimisation technique based on the mechanics of natural selection and genetics. GA's require the parameter set of the optimisation problem to be coded as a finit-length string containing elements (such as 0, 1, #). A population of individuals is created which goes through a process of evolution made up of the principles of combination (cross-overs: swapping chunks of elements between individuals), mutation (changing an element at random) and selection (creating new generations by selection individuals in proportion to their fitness, or 'strength').
The GA is applied after an epoch (of iterations), mating the classifiers and creating new ones. However, "steering" commands are not to be replaced. A solution to this problem is to divide the population of classifiers into a 'replaceable population' and a 'non-replaceable population'. The non-replaceable population is formed by the user or designer (possibly through learning, see §6), consists of classifiers made up by the user or designer and will not be replaced by the GA, although they can be selected for mating. The replaceable population is formed by the genetic algorithm, and is subject to change.
The triggered cover detector operator (TCDO) is activated whenever the CS does not have a classifier which matches a message. It responds by creating a new classifier that covers the message. The action is randomly copied from another action.

In order to keep the rover contingency control up to date with the (new) rule & message system, the entire classifier population should be uploaded after a number of TC's.

5. ON-BOARD ROVER CONTROL

In the normal interactive mode the rover is a simple slave of the human operator. During this interactive mode of operations a command is given and the rover will execute it as good as possible and provide the operator with telemetry. It can be envisaged that the telemetry will be used on-ground to check the effectiveness of the last proposed (by the classifier system) command and so let the classifier system learn from its mistakes and/or achievements.

In a contingency phase the rover autonomy is activated. The activation is be done by an on-board timer when an unexpected time of non-activity (no contact with the ground) is detected. This timer gives the operators on ground the time to solve the problems before the rover takes control. Of course if the problem is solved after the rover switched to autonomous mode the operator can regain control at all times.

Activation of the rover autonomy means activation of the classifier system with the contingency list and latest rules & messages, strengths and environment maps.

During contingency the rover priorities change. It is no longer its priority to reach the target defined by the operator, but it is to survive. The rover will now take into account its housekeeping data to decide what its next move will be. Depending on the contingency the rover will replace its target to a healthy environment, e.g. sudden LOC will provoke the robot to place the target on a cell in the area with positive link status.

The rover assesses what the problem is and proposes a command, no decision can be made, by a human, so the rover will execute the proposed command.

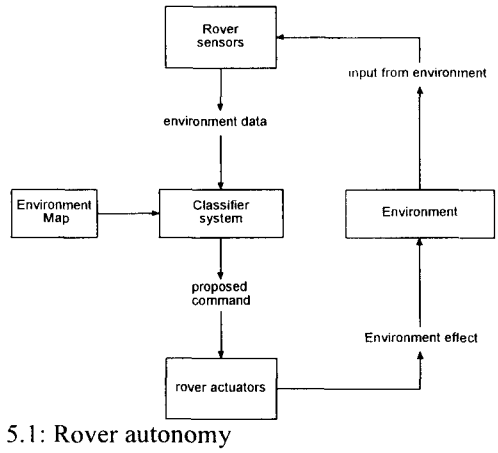

Figure 5.1: Rover autonomy

A choice has to be made to let the rover learn and change the classifiers or not. The classifier system gives the possibility to create new commands in response of TM. By evaluating its decision made earlier, using the current telemetry, the rover could change the strength and classifier rules.
If the rovers learning capacity is unrestricted during a contingency phase, the operator could, if he/she regained control, download the new classifiers to study the newly gained knowledge of the rover. The decision is now up to the operator to let the rover continue with

these classifiers or that the old ones (maybe changed by the operator) will be used.
It has to be studied if the autonomous generated classifiers do not import any uncontrollable rules in the system. For example the rover proposes: 00111010, what will the rover do, move his front left leg backward, move its front right backward, his back left back and forward at the same time and the back right leg does not move?
For this reason it is considered here that the rover does not learn during autonomous mode.
Investigations for a solution to allow generation of combination commands is ongoing, e.g. a first step can be to eliminate command combinations like "normal forward" (01010101 001) and "turn right" (10011001 000) that gives 11011101 001.

The classifier system and in-situ learning (taking into account the result of a command) however can allow the rover to use combinations of the "leg" commands (maybe together with the "steering") to generate new walking methods, e.g. the telemetry indicates the next cell is a slope with loose sand the system will after some re-occurrence of this TM combination generate not a climb command, but a different leg movement, resulting in a better command execution.

6. SIMULATION & RESULTS

The simulation that is done in the framework of this work simulates the learning process of the rover to walk. Starting from simple "leg" movements (see §3.1) and specific situations (defined by the TM) the system proposes a command. The operator evaluates the command and this result is used by the system to give a certain strength to a classifier (in order to get the correct command for that situation, i.e. the correct classifier), finally resulting in "steering" commands (see §3.1).

7. CONCLUSIONS & KEY FEATURES

It can be concluded that the clear separate control during the different modes of operations is simple, robust and safe. The autonomy of the rover is only used when normally the rover system would be lost anyway, so it posses no threat to the planned mission, but it gives extra possibilities to regain the mission.
The concept is flexible enough to allow or not to generate new classifiers when in autonomous mode, no conflict of human vs. rover intelligent system is expected.
The same smart system (classifiers, maps etc.) can be implemented on the rover as on ground.
With the use of the PROLERO rover as a testbed the problem was not approached from the simplest way, a simple 4 wheel rover would probably have made the commands simpler, but this relative complexity in possible commands did not pose any major problems. Although more investigations in un-comprehensible command generation has to be performed.
The use of separate maps containing very limited TM in a binary form together with the concept of target and directions in a map is in line with the goal of simple robust and cheap.

The key features of this paper are: smart & simple control architecture, command generation using classifier systems, learning systems, autonomous rovers.

8. FUTURE DEVELOPMENT

During the realisation of this concept JAQAR engineers discovered some major issues to be resolved, investigations are ongoing in the field of:

- Controlling the generation of non-comprehensible commands during autonomous operations .
- Using the command generation for path-planning and rover operation scheduling purposes.

REFERENCES

[RD1] SPHINcsX, http://www.stanford.edu/~buc/SPHINcsX

[RD2] Legged Locomotion For Planetary Exploration, MSc. Thesis 1st edition Delft University of technology, April 1996, Jasper T. Hillebrand

[RD3] Advanced teleoperation techniques for planetary mobile robots based on fuzzy models, MSc. Thesis 1st edition Delft University of technology, August 1996, Arne Y.J. Matthyssen, and published as ESA WP-1900.

[RD4] 'Evolution in a Microchip' A brief study into Genetic Algorithms; ESTEC ESA-WP 1870, , 1996, Robin Biesbroek.

[RD5] Genetic algorithms in search, optimization and machine learning, Goldberg, Addison-Wesley, 1989.

[RD6] Walking Robots for Planetary Exploration Missions, Martin-Alvares, De Peuter, Hillebrand, Putz, Matthyssen, De Weerd, ISRAM, Montpellier, 1996.