Risk Prediction of a Behavior-based Adhesion Control Network for Online Safety Analysis of Wall-climbing Robots

Daniel Schmidt and Karsten Berns
Robotics Research Lab, Department of Computer Sciences, University of Kaiserslautern, 67663 Kaiserslautern, Germany

Keywords: Genetic Algorithm, Behavior-based, Risk Prediction, Climbing Robot.

Abstract: Risk analysis in combination with terrain classification is a common approach in mobile robotics to adapt robot control to surface conditions. But for climbing robots it is hard to specify, how the robotic system and especially the adhesion is affected by different surfaces and environmental features. This paper will introduce the climbing robot CROMSCI using negative pressure adhesion via multiple chambers, adaptive inflatable sealings and an omnidirectional drive system. It presents the used behavior-based control network which allows the balancing of adhesion force, but fails in extreme situations. Therefore, a risk prediction has been developed which evaluates behavioral meta-data and allows an estimation of current hazards caused by the environment. This prediction is used to perform evasive actions to prevent the robot from falling down.

SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

A general requirement for robots is safety. Commonly, mobile systems have to deal with macro obstacles like persons, furniture, trees or holes depending on their field of application. In these cases the results of a crash and requirements to avoid it can be described well (Kelly and Stentz, 1998) and common approaches of obstacle detection and avoidance can be applied. A more difficult challenge is the adaption to environmental features, which can either not be detected in total or whose impact on the robot is not known sufficiently. Some use methods of terrain classification via simple metrics (Castelnove et al., 2005) or learning methods (Stavens and Thrun, 2006), others try to get more general information about the traversability (Kim et al., 2006) of the surface. These approaches have in common that they collect environmental data, extract key features and derive information which will influence robot navigation.

For wall-climbing robots safety is a main requirement. The problem of terrain analysis is manageable if the climbing system uses legs with independent adhesion units which can test the grip at each foot point (Luk et al., 2001). Other robots use adhesion systems like magnets (Shang et al., 2008) which are safe by default. But for wheeled driving on concrete walls via negative pressure adhesion a prediction of risks is essential. Here, the robot is permanently exposed to a drop-off if it is in motion. Unfortunately, not only the foresighted detection of hazardous features is nearly impossible due to missing sensor accuracy and/or limited payload. Also the impact of features like surface roughness, sheathing defects, porous areas or micro channels on the adhesion system can not be described sufficiently (in contrast to macro features).

This paper presents a risk prediction method and suitable measures to avoid them. Upcoming section 2 will introduce some fundamentals. Section 3 presents the procedure of risk prediction, section 4 shows how the needed parameters are determined via training. The experimental results and safety measures are presented in section 5, conclusions follow in section 6.

2 FUNDAMENTALS

The research presented in this paper is aimed at the *climbing robot* CROMSCI (Schmidt et al., 2011) but works for similar systems, too. CROMSCI is designed to be used for inspections of large concrete buildings as depicted in figure 1. Requirements for this task are a relatively high velocity even in vertical direction or overhead for a sufficient fast navigation between inspection points and the ability of carrying a high payload in terms of inspection sensors. The most innovative feature of CROMSCI is the negative pressure system consisting of seven individual adhesion chambers which allow a balancing of downforces. For high maneuverability and fast continuous motion it is

Schmidt D. and Berns K..
Risk Prediction of a Behavior-based Adhesion Control Network for Online Safety Analysis of Wall-climbing Robots.
DOI: 10.5220/0003974701180123
In *Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2012), pages 118-123
ISBN: 978-989-8565-21-1
Copyright © 2012 SCITEPRESS (Science and Technology Publications, Lda.)

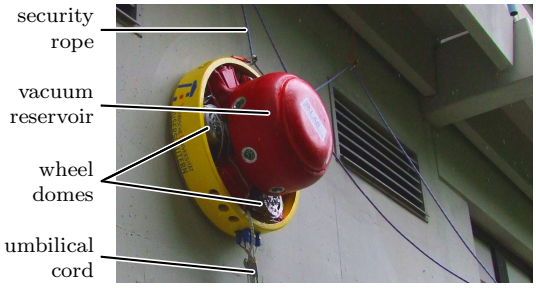

Figure 1: Climbing robot CROMSCI at a concrete wall.

equipped with three unsprung steerable driven standard wheels. A load cell is integrated into each wheel to measure forces and torques at the wheel's contact point. CROMSCI can be equipped with a movable manipulator arm to carry inspection sensors. For communication and energy supply it is connected to a ground station via an umbilical cord. Some key data of the robot are a maximum velocity of 9.81 m/min, 80 cm diameter, a weight of 45 kg and an additional payload of about 10 kg.

The control software of CROMSCI makes use of the *behavior-based control architecture* iB2C (Proetzsch et al., 2010) on all abstraction levels reaching from closed-loop control up to high deliberative functions. In general an iB2C behavior is an algorithmic element which generates control data based on its inputs. All behaviors share a common meta data interface for interaction. Two of these data ports deliver information about the current state of the behavior and are important in this context: *activity* $\vec{a} \in [0,1]^w$ shows the real amount of action the behavior performs whereas the *target rating* $r \in [0,1]$ represents its satisfaction with the current situation.

The *adhesion control system* itself consists of a network of 47 of these behavior elements as published in (Schmidt and Berns, 2011). The lower part of the network consists of the chamber controllers which perform the closed-loop pressure control. Their meta values are presented exemplarily in equations 1 and 2. The behavior's *activity* a_{CC} depends on the actual valve area A^{act} and its maximum A^{max} (the larger the valve opening the more active). The *target rating* r_{CC} uses the control difference of desired p^{des} and actual chamber pressure p^{act} compared to a maximum difference Δp^{max} and therefore is unhappy if the desired pressure value can not be reached or if it is not activated (ι_{CC} is an internal activation value).

$$a_{CC} = \iota_{CC} \cdot \frac{A^{act}}{A^{max}} \tag{1}$$

$$r_{CC} = \max\left(0, \min\left(1, \frac{|p^{act} - p^{des}|}{\Delta p^{max}} + (1 - \iota_{CC})\right)\right) \tag{2}$$

In an outer control loop the downforce is adjusted. The meta values of all behaviors are calculated in a similar way as for the chamber controllers. Additional behaviors analyze the robot state or inhibit chamber controllers in cases of high leakages to prevent the complete adhesion system from fail. Nevertheless, the optimal downforce is not easy to determine because the robot neither should fall down nor get stuck. Even if it is in the ideal range at about 2000 N there still exists the chance of the robot to slip or tilt which could result in a drop off. Some additional measures like a traction control system have already been developed to reduce these effects but can not prevent them completely (Schmidt et al., 2011).

3 RISK PREDICTION

The basic control measures work in general, but are not able to prevent the robot from a drop-off in certain situations. Although the robot is equipped with a light-weighted Hokuyo laser ranger for obstacle avoidance, these external sensor data have a relatively low accuracy compared to the micro-features which need to be detected for a foresighted evaluation of the terrain. Therefore, internal sensors like the pressure sensors have to be taken into account here. This approach is possible because of the redundant multiple chamber system which allows the failure of some chambers for a short period of time without endangering the system. In practice the front chambers in driving direction are exposed to hazardous features first which allows a judgement of the upcoming terrain characteristics. First experiments have proven that pressure values itself are not sufficient for risk prediction. The idea is to evalute *activity* and *target rating* values of the adhesion behaviors instead. Especially the different *target ratings* provide information about the system state because they represent individual satisfaction values of controllers.

The intention is now to receive a risk value of an *evaluation function* $E(\vec{a},\vec{r})$ which is one or above if the robot will drop off within the next seconds (if no evasive action is performed). Of course, this risk value should indicate a potential drop off early enough to allow evasion actions. On the other hand it must stay below one if the robot adhesion is not endangered to avoid false positives.

$$E(\vec{a},\vec{r}) = \sum_{i=0}^{n-1} w_{a_i} \cdot a_i + w_{sa_i} \cdot s(a_i) + w_{r_i} \cdot r_i + w_{sr_i} \cdot s(r_i) \tag{3}$$

The current approach uses a weighted sum (equation 3) $E(\vec{a},\vec{r}) : [0,1]^{4n} \mapsto \mathbb{R}$ as evaluation function

based on the meta data of the n behaviors. At this juncture *activity* and *target rating* values a_i respective r_i of behavior i are used in combination with corresponding weights w_{a_i} and w_{r_i}. In addition also low-pass filtered meta values $s(a_i)$ and $s(r_i)$ with $s(x) = 0.3 \cdot x + 0.7 \cdot s'(x)$ and corresponding weights are taken into account which reduces peaks in the evaluation function. It is also possible to calculate and use other values like average, median or variance in the same way. Recent experiments have shown that these additional values may allow a better prediction, but this enhancement comes with two restrictions: At first one needs to determine a lot of more weights, at second there is a higher specialization to certain situations and parameters like vehicle velocity. The main problem is now to determine the optimal weights $\vec{w} \in \mathbb{R}^{4n}$. It is obvious that this large number of possibilities can not be set by hand. The next problem is that a forecast of an adhesion failure is only possible if one knows in which situations the system will fail.

Therefore, a learning method with training data needs to be applied to find suitable weights. At first one needs a measure if the robot fails in a situation or not and determine important characteristics. The identification of a drop off is done by an *adhesion score function* $S_A(F_z, x_F, y_F) : \mathbb{R}^3 \mapsto [0,1]$ which uses two different indicators as given in equation 4:

$$S_A(F_z, x_F, y_F) = \max\Big(S_{A_F}(F_z), S_{A_P}(x_F, y_F)\Big) \quad (4)$$

The first indicator is the *current downforce* value F_z (equation 5) measured by the embedded load cells. If F_z is below threshold F_z^{min} the robot falls down:

$$S_{A_F}(F_z) = 1 - \max\left(0, \min\left(1, \frac{F_z - F_z^{min}}{F_z^{max} - F_z^{min}}\right)\right) \quad (5)$$

The second measurement unit is the *point of downforce* which describes the chance of robot tilt (equation 6). If the distance of the downforce point with coordinates x_F and y_F from the robot center is too large (above a threshold d^{max}) the robot drops off. The used threshold values depend on system parameters like weight, wheel distance or friction coefficient.

$$S_{A_P}(x_F, y_F) = \max\left(0, \min\left(1, \frac{\sqrt{x_F^2 + y_F^2}}{d^{max}}\right)\right) \quad (6)$$

Independent from the type of learning algorithm one needs some *experimental training data*. To get this data, the robot has to be faced with situations in which the adhesion system reaches its limits and the robot falls off ($S_A = 1$) as well as situations which are harmless or still manageable by the system ($S_A < 1$). Each training set is a double array consisting of a time value $t \in \{0, m-1\}$, all meta data from the considered behaviors and the adhesion score[1] $S_A(t)$ at that timestep. Of course, the size m of the tables varies from one dataset to another whereas the setup of behaviors has to be fix.

In literature different learning and optimization methods exist which can be used in general. Therefore one needs to find a suitable approach to extract the needed weight values out of the training data. *Artifical neural networks* e. g. are a classic method for pattern recognition but will not fit in here because of missing input-output samples. Another approach is *reinforcement learning* which tries to optimize a problem via trial-and-error. Nevertheless it is more linked to a mapping from situations to actions than for the given problem. *Simulated annealing* might be an appropriate approach, but it does not seems to be a good idea to follow and optimize only one solution in the present case. Therefore, the principle of *genetic algorithms* is applied to determine the best weights (Gerdes et al., 2004). The idea is to update the evaluation weights randomly until the desired performance is achieved.

4 GENETIC ALGORITHM

As usual one needs a population $P(s)$ of individuals at step s. Each individual has a chromosome c with genes which can mutate randomly in a predefined way to optimize the desired function. In this case one individual consists of a vector of weights $\vec{w} = (w_{a,0}, w_{r,0}, w_{sa,0}, \ldots w_{sr,n-1}) \in [-1,1]^{4n}$ which is used for the weighted sum of n behaviors as shown before. At the beginning of the process a set of individuals is created with random genes. In each optimization step as illustrated in figure 2 the fitness $F(c)$ of all individuals is calculated which describes the chance of an individual to survive. The next generation $P'(s)$ is created by a selection of original individuals which are updated with certain probabilities via genetic operations as described in section 4.2.

4.1 Evaluation of Individuals

Of course, the principle of behavior evaluation is not limited to the given problem. However, the fitness function is the most difficult part since it describes the optimization problem and has to be set properly to achieve the desired results. In this case an individual is good if the evaluation function E (with the

[1] From now on the results of functions will be shortened like $S_A(t) = S_A(F_z(t), x_F(t), y_F(t))$ for clarity.

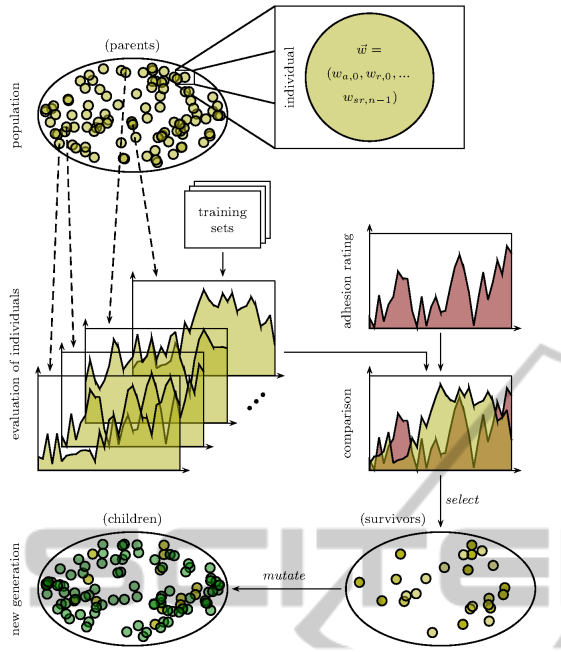

Figure 2: One evolution step with current population, selection, mutation of survivors and the next generation.

weights) is a good prediction of the adhesion score S_A in equation 4. As measuring unit, a rating function R_E is used which compares the evaluation result of an individual with the adhesion score. At first, the weights are used to calculate the evaluation value according to equation 3 for each timestep t of one training set. One receives a list of evaluation values $E(t)$ over time which have to be compared to the corresponding adhesion score $S_A(t)$. The *rating of weight evaluation* R_E is done according to equation 7 and calculated for each combination of individual and training set.

$$R_E = -\sum_{t=0}^{m-1} P_{E\Delta}(t) - \sum_{t=0}^{k} P_E^{unw}(t) - M_E^{unw} - \sum_{t=k+1}^{k+\Delta t} P_{E_1}^{des}(t) - \sum_{t=k+\Delta t+1}^{m-1} P_{E_2}^{des}(t) - M_E^{des} \tag{7}$$

This rating considers three aspects: At first, the evaluation function $E(t)$ should *stay below* the adhesion rating $S_A(t)$, otherwise the rating value is diminished by a penalty $P_{E\Delta}$ according to equation 8. The used functions and constants have been determined carefully for an optimal and balanced rating function.

$$P_{E\Delta}(t) = \begin{cases} ((E(t) - S_A(t)) \cdot E(t))^3 & \text{, if } E(t) > S_A(t) \\ 0 & \text{, else} \end{cases} \tag{8}$$

The second aspect is the *avoidance of unwanted values* which produce false alarms. This is split up into a penalty based on the differences of evaluation and adhesion rating P_E^{unw} according to equation 9 and a basic mallus M_E^{unw} if at least one undesired value exists (equation 10). S_A^{haz} denotes a threshold for a hazard which is set to 0.9, 10^{10} and 10^9 are constants.

$$P_E^{unw}(t) = \begin{cases} \left((E(t) - S_A^{haz}) \cdot \max\left(E(t), S_A(t)\right)\right)^3 \cdot 10^{10} & \text{, if } E(t) > S_A^{haz} \\ 0 & \text{, else} \end{cases} \tag{9}$$

$$M_E^{unw} = \begin{cases} 10^9 & \text{, if } S_A(t) < S_A^{haz} \forall t \in [0,k] \wedge \exists E(t) \geq S_A^{haz},\ t \in [0,k] \\ 0 & \text{, else} \end{cases} \tag{10}$$

The value k depends on the type of training set: If the adhesion rating stayed below S_A^{haz} the complete dataset is processed here (equation 11), otherwise it considers only the timespan to the timestep t^{haz} at which the adhesion rating reached a hazardous value minus the double time Δt. This describes the desired timespan of the reaction time with $\Delta t \leq t^{react} \leq 2 \cdot \Delta t$.

$$k = \begin{cases} m-1 & \text{, if } S_A(t) < S_A^{haz} \forall t \in [0, m-1] \\ t^{haz} - 2 \cdot \Delta t & \text{, else} \end{cases} \tag{11}$$

In the same way a mallus and a penalty for *missing desired values* are applied, if the adhesion rating S_A of this data set is above the threshold at least once (so that $k < m-1$). The evaluation $E(t)$ has to reach 1 within the range $[k+1, k+\Delta t]$ for prediction, otherwise mallus $M_{E_1}^{des}$ (equation 14) is added as well as penalty $P_{E_1}^{des}$ in equation 12. Penalty $P_{E_2}^{des}$ from equation 13 tries to push the evaluation function above 1 over the remaining time steps until $m-1$.

$$P_{E_1}^{des}(t) = \begin{cases} (1 - E(t))^3 \cdot 10^{10} & \text{, if } E(t) < 1 \wedge E(t) < 1 \forall t \in [k+1, k+\Delta t] \\ 0 & \text{, else} \end{cases} \tag{12}$$

$$P_{E_2}^{des}(t) = \begin{cases} (1 - E(t))^3 \cdot 10^6 & \text{, if } E(t) < 1 \\ 0 & \text{, else} \end{cases} \tag{13}$$

$$M_E^{des} = \begin{cases} 10^9 & \text{, if } E(t) < 1 \forall t \in [k+1, k+\Delta t] \\ 0 & \text{, else} \end{cases} \tag{14}$$

If p training sets are used the mean-square average of the ratings is determined (equation 15):

$$R = -\sqrt{\frac{1}{p} \cdot \sum_{i=0}^{p-1} \left(R_{E_i}^2\right)} \tag{15}$$

4.2 Fitness, Selection & Mutation

Based on the final rating value R the *fitness* $F(c)$ of an individual can be determined. Equation 16 shows this calculation based on minimum and maximum values for the rating value R^{min} and R^{max} (which can either

be set fix or dynamic based on lowest and highest actual rating values) and on a basic fitness F^{bas} for all $|P|$ individuals of that population. The chance $p(c)$ of an individual c to survive depends on the ratio of individual to population fitness (equation 16, right).

$$F(c) = \frac{\frac{R(c)-R^{min}}{R^{max}-R^{min}} + \frac{F^{bas}}{|P|}}{1+\frac{F^{bas}}{|P|}}, \quad p(c) = \frac{F(c)}{\sum_{i \in P} F(i)} \quad (16)$$

Since the exploration of the search space is most important, crossover operations are not considered here: *"If optimality is sought, crossover may be deleterious"* (Spears, 1993, page 231). In fact three different *mutation types* are used for adaption: A weight can be updated with a small *random offset*. The used probability p_m^{off} is 0.75 that one weight of an individual is adapted. Furthermore, an update with a new *random weight* with probability $p_m^{rand} = 0.5$ is possible. Additionally all weights of an individual can be changed with a *random multiplication factor* $f \in [0.9, 1.1]$. The probability p_m^{mul} that one individual is updated inside of the population is 0.1.

Since the values can skitter away it is useful to keep the best individual inside of the population. On the other side the population should get the chance to expand in all directions. Therefore, this approach additionally uses a kind of *elitism function* in a way that there exists a certain chance that the best individual (that has been found in the past) will be injected into the population again. Additionally, this chance decreases over time if no better individual could be generated to enlarge the evolution space and to reduce the effect of local minima. The breakpoint R^{bp} of the learning procedure is reached if no rating R_{E_i} of the best individual gained a mallus M_E^{unw} or M_E^{des}. So far, the algorithm is stopped manually.

5 RESULTS AND MEASURES

Some exemplary results are given in figure 3. Here the adhesion scores S_A (gray) of two different data sets and final evaluation values E (black) which are limited to $[0,1]$ are shown. In the given experiment 10 training sets have been used to determine a set of 90 weights of 45 considered behavior values. In all cases the robot was driven down a wall but at different positions reaching from even and rough surfaces to patches with deep grooves. The learning algorithm using 100 individuals was able to train weights which guarantee a certain reaction time t^{react} and avoid false-positives in the training sets. Figure 4 shows the increasing evaluation values over time which are better in cases of small populations ($|P| = 100$, black)

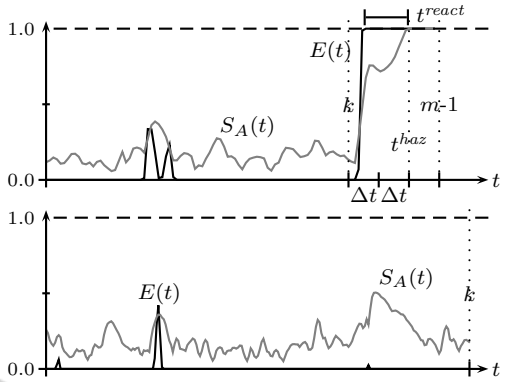

Figure 3: Example for desired results after 1900 evolution steps (approx. 1330 s): If $S_A(t)$ reaches one $E(t)$ should signal this beforehand (top).

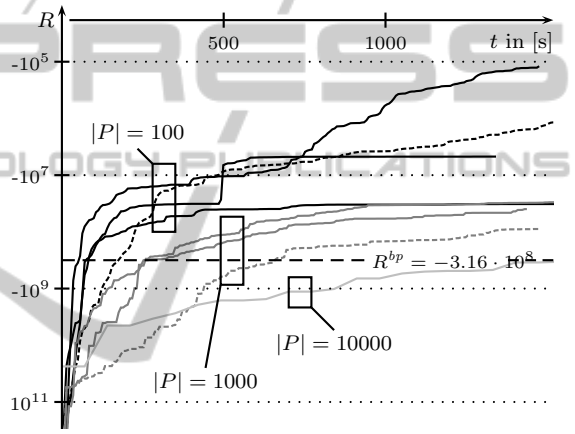

Figure 4: Enhancements of the rating value R of the best individual over time with different population sizes $|P|$.

performing a faster evolution step compared to larger populations. The dashed lines indicate experiments without random multiplication ($p_m^{mul} = 0.0$) with a slower convergence at the beginning.

To evaluate the learning results the robot again performs similar trajectories on the structured surface with defects and cracks. Figure 5 shows reaction times t^{react} between detection ($E(t) = 1$) and drop off ($S_A(t) = 1$) while the robot tries to handle the deep cracks. Again, evaluation E should signal a drop off early enough to have enough time for counteractive measures. The black bars indicate a more uniform crack in contrast to a complex crack structure (gray bars). In total, 31 test runs on a cracked structure have been executed with only one false-negative, two with a too short reaction time below 0.5 s and five runs with a reaction time larger than 3 s. Further experiments have shown, that the behavioral situation is completely different if the robot drives upwards so another set of weights has to be trained for this case.

Beside the correct detection also the avoidance of

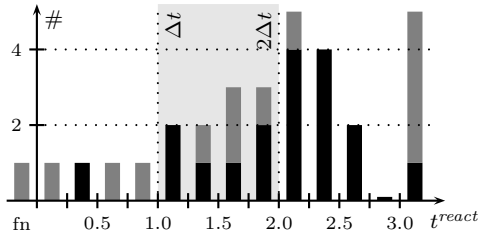

Figure 5: Reaction time t^{react} in $[s]$ and one false-negative (fn) of examples while facing different cracks.

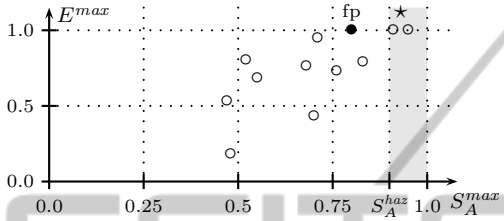

Figure 6: Maximum evaluation and rating values on different rough terrains, ⋆ marks tolerated false-positives (fp).

false-positives is important. Therefore, the reaction on rough terrain with defects has been tested which do not lead necessarily to a drop-off. Figure 6 shows 12 test runs with only one false-positive (black circle) and two detections ($E^{max} = 1$, marked with ⋆) which are tolerated because of $S_A^{max} > S_A^{haz}$. In practice, the evaluation system has to be trained once and can be applied to similar situations and setups.

Each detection of safety-critical situations is useless without *counteractive measures*. So far, a reversed replay of the robot trajectory is implemented. The responsible behavior has been embedded into the control system and is stimulated, if the evaluation function E reaches a value of 1. In this case, the current driving operation is cancelled and the last commands are countervailed. The idea is that the way was not dangerous so far so the robot should drive back the same trajectory until a safe position has been reached and the adhesion system can recover.

6 CONCLUSIONS

This paper presented a risk prediction approach for wall-climbing robots. Based on training data a genetic algorithm is used to find suitable weights for a general evaluation function which is used here to predict an upcoming drop-off. Experiments have proven the functionality of the approach and the benefit for robot safety. The next step is to adapt the prediction system to be able to handle different situations (e. g. driving up or down) which need to use differing sets of weights since one-fit-all-weights do not exist.

REFERENCES

Castelnove, M., Arkin, R., and Collins, T. R. (2005). Reactive speed control system based on terrain roughness detection. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 891–896.

Gerdes, I., Klawonn, F., and Kruse, R. (2004). *Evolutionäre Algorithmen*. Vieweg Verlag, Germany, 1. edition.

Kelly, A. and Stentz, A. (1998). Rough terrain autonomous mobility - Part 1: A theoretical analysis of requirements. *Autonomous Robots*, 5(2):129–161.

Kim, D., Sun, J., Oh, S. M., Rehg, J. M., and Bobick, A. F. (2006). Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In *IEEE International Conference on Robotics and Automation (ICRA) 2006*, pages 518–525.

Luk, B. L., Cooke, D. S., and others (2001). Intelligent Legged Climbing Service Robot For Remote Inspection And Maintenance In Hazardous Environments. In *8th Conference on Mechatronics and Machine Vision in Practice*, pages 252–256.

Proetzsch, M., Luksch, T., and Berns, K. (2010). Development of complex robotic systems using the behavior-based control architecture iB2C. *Robotics and Autonomous Systems*, 58(1):46–67.

Schmidt, D. and Berns, K. (2011). Behavior-based adhesion control system for safe adherence of wall-climbing robots. In *14th International Conference on Climbing and Walking Robots (CLAWAR)*, pages 857–864.

Schmidt, D., Hillenbrand, C., and Berns, K. (2011). Omnidirectional locomotion and traction control of the wheel-driven, wall-climbing robot, Cromsci. *Robotica Journal*, 29(7):991–1003.

Shang, J., Bridge, B., Sattar, T., Mondal, S., and Brenner, A. (2008). Development of a climbing robot for inspection of long weld lines. *Industrial Robot: An International Journal*, 35(3):217–223.

Spears, W. M. (1993). Crossover or mutation. In *Foundations of Genetic Algorithms*, volume 2, pages 221–237.

Stavens, D. and Thrun, S. (2006). A self-supervised terrain roughness estimator for off-road autonomous driving. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 469–476.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS