Human-in-the-Loop Control through Kinematic Redundancy Resolution for Space Exploration Rovers

Velin Dimitrov and Taşkın Padır
Electrical and Computer Engineering
Northeastern University
360 Huntington Ave
Boston, MA 02115
{v.dimitrov, t.padir}@northeastern.edu

***Abstract*—We present an architecture to enable the modeling of human-in-the-loop control problem of space exploration robotic systems. We describe a shared control architecture, originally developed for assistive and disaster response robotics, for potential use in a space exploration scenario. Examples relevant to space are provided to highlight the different elements of the shared control architecture. A method for integrating human input with the autonomous behavior of the system using redundancy resolution taking into account primary and secondary tasks is presented. Five different modalities of shared control are described with the redundancy resolution method. The shared control architecture and redundancy resolution allow human input to be integrated as constraints on the kinematic model of the system.**

TABLE OF CONTENTS

1. INTRODUCTION

This paper introduces a human-in-the-loop control framework for enabling a small operations team to control semi-autonomous rovers capable of efficiently locating, identifying, and retrieving various geologic samples. As humankind's exploration capabilities on the Moon and Mars continue to improve [1], and as missions evolve to include multiple robotic platforms (including rovers, micro-rovers, hoppers, and orbiters) with varying complexity and resources, a paradigm shift in mission design approach is required to rely more on the autonomy of these systems. We present the design of an adaptable, reconfigurable, and resource-aware shared control framework for sample return exploration missions with a heterogeneous human-robot team. Such a control framework will not only enhance our ability to explore the Moon and Mars autonomously and more efficiently but also enable the exploration of celestial bodies farther than Mars without relying on robust, low-latency communication channels. It is possible to classify most robots that are currently deployed in applications in two categories: (i) fully autonomous robots performing specific tasks [2–4], and (ii) teleoperated robots with little high-level intelligence [5–7].

978-1-4673-7676-1/16/$31.00

We acknowledge that not all human-robot interaction fall into these two categories but they represent a wide majority of systems that are currently in use. As we attempt to close the gap in between these two classes, new control techniques are needed to dynamically shift the level of control between the human operator and the intelligent robot resulting in a true shared control system.

The framework design, succinctly summarized in Figure 1, is novel in three key aspects. (i) It is adaptable as it dynamically evaluates the capabilities of the robot platforms and human resources that make up the heterogeneous human-robot team. (ii) The framework is reconfigurable, designed to be robust to the addition and removal of human-robot team members, and as a result it is inherently fault-tolerant to individual system failures. (iii) The framework is resource-aware in terms of hardware and software requirements of the mission tasks provided by the human operator at the supervisory level and as a result allocates the resources to best achieve the given exploration task.

The paper is organized as follows: Section 2 introduces the shared control framework adapted from [8], the elements are the same but their importance and function is different in a space exploration mission, Section 3 summarizes the kinematics and redundancy resolution from [9] used to integrate the human operator input with the semi-autonomous system, Section 4 describes the new method for integrating the human input as a constraint on the kinematics equations governing the system as primary and secondary tasks, and finally Section 5 discusses the future work to follow and concludes with a summary of the main contributions.

2. SHARED CONTROL ARCHITECTURE

Figure 1 shows the architecture originally developed for assistive robotics and disaster response robotics in [8]. The shared control architecture is easily adaptable and extensible to grow as research thrusts push in different directions. In this section, we describe the basic elements of the architecture, which are unchanged from before. The main difference though is that given the significant challenges of communication in space environments and emphasis on reducing risk since repair missions are essentially impossible, certain elements of the architecture are more important and serve different purposes than in [8]. We provide examples for each element in a hypothetical space exploration mission.

The structure of the architecture is split into quadrants related to *where* (cyber and physical) each component resides and *who* (human and robot) contributes the information in each component. The components in the cyber realm operate in an abstract area that could be in the internet (on a cloud

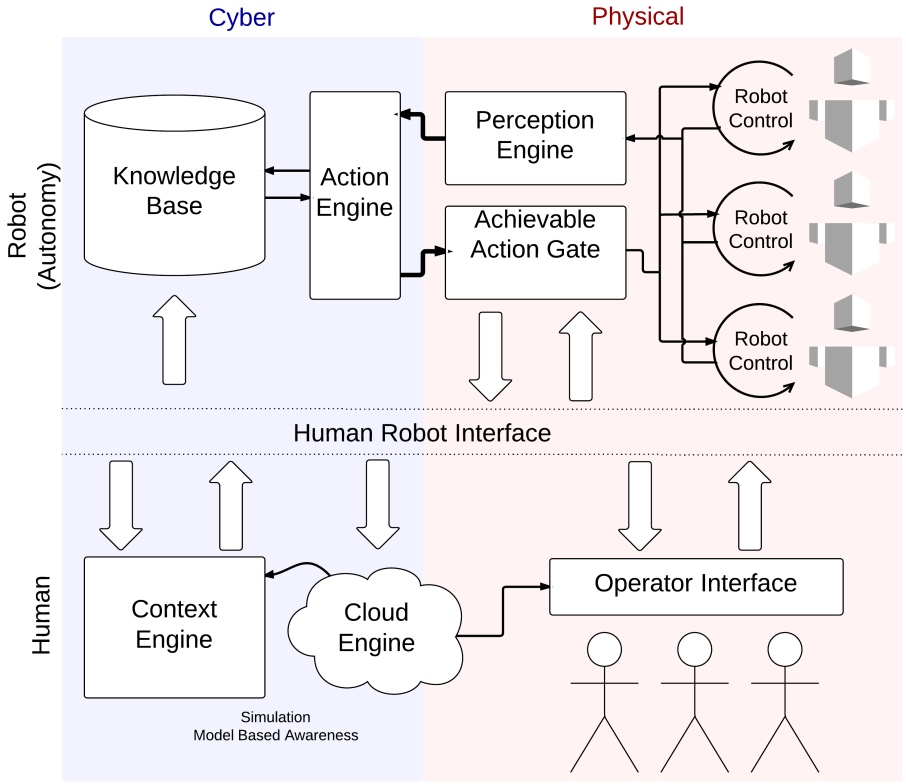

Figure 1. The shared control architecture consisting of the knowledge base, action engine, achievable action gate, perception engine, low-level robot control, human robot interface, context engine, cloud engine, and operator interface. The architecture is split in both cyber and physical domains, and also human and robot domains depending on *where* elements are located and *who* contributes information to each element.

platform for example), could be on the robot, or could be distributed across several different areas. The cyber realm is not characterized by physical location but the availability of significant computation power and bandwidth within the realm, where heavy computing and intelligence can be easily implemented. The components in the physical realm are associated with the tangible objects of the system such as the on-board computing of the systems and interfaces the operators utilize.

A second categorization splits the architecture between components that directly utilize mostly human-centric information or robot-centric information. The components base their outputs on the information that is available, so this is a natural dichotomy based on how the control within the system is split between the autonomous agents and the human-driven ones. The human robot interface provides an abstraction for the information transfer between the two realms encompassing information generated by the operators and also automatic contextual information from the other elements. It is separate, but includes information from the operator interface. In reality, these boundaries are fuzzy and not clear-cut defined in a real system, but such a categorization helps with the comprehension of information sources and flow within the system.

The *Knowledge Base* stores global strategies and approaches for the system, providing options to the action engine further down the line. The knowledge base is aware of the history of the action engine, and provides the high-level abstract goal that the system should try to achieve along with which control modality is most appropriate to accomplish the given task. For space systems, the knowledge base operates on an infrequent and asynchronous time frame, updating all but its most critical information only when high-gain antennas are available for large information transfer. Because of this, the knowledge base can store an immense amount of information and has the time to search through it to create a relevant plan for the action engine. In addition to the history of the action engine, the knowledge base receives information from the context engine to adjust global strategy based on the context of the environment, human operator, and system model. The knowledge base is critical in a space environment because it can guide the system to choose the correct plan of action before the information for the decision has even made it back to the operators.

The *Action Engine* takes the global strategy and plan given by the knowledge base and generates a set of potential actions that the robots can take to achieve the tasks. The action engine has access to the information from the perception engine, which provides limited information on the state of the robot, and passes its desired actions to the achievable action gate. The action engine may be completely in the cyber realm or may be infringing in the physical realm as well, especially if it is distributed in terms of computing. The action engine is where the different control modalities are implemented. We discuss in section 4 one implementation for the control modalities utilizing the redundancy inherent in the system.

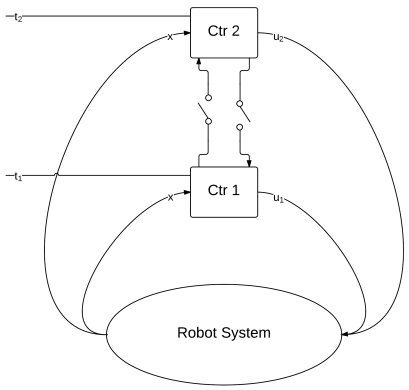

Figure 2. A proposed architecture for the robot controllers. Each robot may have different controllers providing different inputs at different time-scales. In addition, there may be coupling between the controllers as well. The architecture allows for both parallel and series combinations of controllers to be modeled.

The *Perception Engine* has a high-bandwidth, low-latency link with the robot sensors collecting and sorting the information from those sensors. The primary purpose of the perception engine is to interpret the sensor information and pass along only the information relevant to generating a set of desired actions to the action engine. Any system-wide state estimation (not robot-specific state estimation) should occur in the perception engine since it is the first place that information from all the robots is aggregated. The perception engine is where information about the environment from the rover, any orbiting assets, or even potentially hopper rockets surveying ahead of the robot can be aggregated to provide the best situational awareness possible to the system.

The *Achievable Action Gate* takes the desired actions provided by the action gate and checks if they are actually achievable and relevant for the system. For example if the action engine requests a set of actions that are out-of-bounds for a given robot, the action gate will modify those actions so they are as close to the desired ones as possible but within the capabilities of the robot. The achievable action gate is where the operator input from section 4 is fused with the autonomous behaviors of the system.

The *Robot Control* block accepts the actions from the action gate and translates them to low-level commands that the motion controllers on each robot can accept and accomplish. Figure 2 show the proposed architecture for modeling robot controllers. Individual controllers can provide inputs directly to the system or go through another controller, enabling it model both parallel and series combinations of controllers. t_1 and t_2 provide independent triggers that enable the controllers to run on different timescales.

The *Cloud Engine* receives limited state information about the robot system through the human robot interface and passes the information to the cloud, harnessing the combination of significant human and computational resources, to provide simulations and model-based awareness and control improvements. For example, since there is significant time delay in communicating with space robot, a model of the robot can be generated in the cloud engine and presented to the operators. The cloud engine, like the action engine, can straddle the grey boundary between the cyber and physical realm.

The *Context Engine* receives information from the human-robot interface similar to the cloud engine to understand the environment and situation around the robots. In addition, it can utilize the information from the cloud engine to help decipher the information and context in an accurate and efficient manner. The contextual information is sent to the knowledge base to be recorded and included in the plan later sent to the action engine. The context engine is where the most relevant information about the environment around the robot, gathered by the information coming back, can be generated and presented to user.

Finally, the *Operator Interface* serves as the carrier of information between the robots and the operator or potentially multiple operators. The interface can change based on the information provided by the cloud engine. Model-based awareness algorithms running in the cloud engine can change the perspectives of the information presented on the operator interface. All information through the operator interface does not necessarily need to be validated by the operator always, but it is available for introspection if need.

Despite the simple nature of the diagram, we feel this is a preliminary implementation of a very powerful approach to model shared control of space exploration systems that has potential impacts across the whole range of application domains. It can enable the cross-pollination of ideas and approaches, the ability to compare and contrast ideas in common structure, and allow researchers to describe shared control systems in a common language. One of the central challenges of the framework is the integration of the human input with the autonomous behaviors across the human robot interface inside the action engine and achievable action gate. In the next sections, we describe the theory behind one approach for enabling shared control by integrating the human inputs as constraints on the kinematics of the system.

3. REDUNDANCY RESOLUTION

For the sake of simplicity, we will demonstrate the concepts with a simple 5-DOF example, a robot whose base is constrained to the plane (3-DOF) with a 2-DOF arm consisting of a panning and tilting shoulder joint. The end-effector of the arm can be off the plane that constrains the robot base. The variable name conventions and general methodology follows the formulation in [9] and the key equations are reproduced here for easy reference. Because of the vector and matrix representations, the concept scales easily to higher DOF systems. The pose of the robot and the joint angles of the robot form a set of generalized coordinates in Eqn. 1, where the terms are lateral displacement, forward displacement, robot orientation, and the two joint angles of the arm respectively.

$$\eta = \begin{bmatrix} x_v & y_v & \theta_v & \theta_1 & \theta_2 \end{bmatrix} \tag{1}$$

The task space coordinates are accordingly in Eqn. 2. The lateral displacement, forward displacement, and orientation are trivial given the robot is constrained to the plane. The z_q term is only used to describe the height of the end-effector on the arm above the ground, since the arm is not constrained to the plane.

$$p_q = \begin{bmatrix} x_q & y_q & z_q & \theta_q \end{bmatrix} \tag{2}$$

The rest of the section highlights the important results of the derivation from [9] that we utilize to implement the shared control through the redundancy inherent in the system. The kinematic constraints can be summarized in a very compact form reproduced in Eqn. 3.

$$A(\eta)\dot{\eta} = 0 \tag{3}$$

Depending on the specific task we want to accomplish, the 5-DOF system may be redundant and we will demonstrate several examples of shared control implemented by introducing pseudovelocities that make use of the redundancy in the system. Pseudovelocities are formed by the transformation matrix $B(\eta)$ such that

$$\dot{\mu} = B(\eta)\dot{\eta} \tag{4}$$

where $\dot{\mu}$ is the pseudovelocity vector and the matrix B is chosen such that $\begin{bmatrix} A^T & B^T \end{bmatrix}^T$ is invertible. So, we can write,

$$\begin{bmatrix} A \\ B \end{bmatrix}^{-1} = \begin{bmatrix} \Sigma & \Gamma \end{bmatrix} \tag{5}$$

where Σ and Γ satisfy the following conditions: $A\Sigma = I_3$, $B\Sigma = 0$, $A\Gamma = 0$, and $B\Gamma = I_5$.

We resolve the redundancy in the system through the Moore-Penrose pseudoinverse where ζ is the null-space vector.

$$\dot{\eta} = \Gamma\dot{\mu} = \Gamma(J_q\Gamma)^{\dagger}\dot{p}_q + \Gamma\{I - (J_q\Gamma)^{\dagger}(J_q\Gamma)\}\dot{\zeta} \tag{6}$$

If we want, we can accomplish a primary task that is of lower dimensionality, and simultaneously specify a secondary task to implement simultaneously [10]. We will utilize this formulation to represent the human input to the shared control system as either part of the primary or secondary task. Let's assume that we can formulate the primary and secondary task by relating the pseudovelocities to the task space points that correspond to the task through the appropriate Jacobian matrix.

$$\dot{p}_p = J_p(\mu)\dot{\mu} \tag{7}$$
$$\dot{p}_s = J_s(\mu)\dot{\mu} \tag{8}$$

Then the equivalent result to Eqn. 6 including both the primary and secondary tasks becomes,

$$\begin{aligned}\dot{\eta} =& \Gamma\{J_p^{\dagger}\dot{p}_p + (I - J_p^{\dagger}J_p)\tilde{J}_s^{\dagger}(\dot{p}_s - J_sJ_p^{\dagger}\dot{p}_p) \\ &+ (I - J_p^{\dagger}J_p)(I - \tilde{J}_s^{\dagger}\tilde{J}_s)\dot{\zeta}\}\end{aligned} \tag{9}$$

Eqn. 9 represents the generalized velocities required to accomplish both the primary and secondary tasks through the redundancy resolution. By formulating the tasks in Eqns. 7 and 8, we can implement different shared control modalities and using Eqn. 9 we can tell the robot in joint space how to achieve the tasks.

4. SHARED CONTROL

In [11], Zermelo's navigation problem is used as an example to consider a range of methods as to how shared control may be implemented including traded control, indirect shared control through cues, coordinated control, collaborative control, virtual constraint, and blended shared control. This categorization is very thorough and covers a wide range of possible shared control modalities. We use these categories as a template to show how each shared control modality, with the exception of indirect control through cues, can be implemented using the primary and secondary task allocation from the previous section. We leave out the cue-based control modality because it involves modeling the "internal" control loop of the human response, and that would be better accounted for in a dynamical model.

Traded control involves having the autonomous agent and human agent trading off control of the platform on a time-based or event-based trigger. The system can either be in full autonomous mode or under complete teleoperation, with no intermediate option. This can be simply expressed with a couple of cases,

$$\dot{p}_p = \begin{cases} \dot{p}_u \text{ when } t_1 = 1 \text{ and } t_2 = 0 \\ \dot{p}_a \text{ when } t_1 = 0 \text{ and } t_2 = 1 \end{cases} \tag{10}$$

where t_1 and t_2 correspond to the time or event triggers on the parallel control loops shown in Fig. 2 allowing the robot control block to transfer between teleoperation and fully autonomous operation. $\dot{p}_u$ and $\dot{p}_a$ correspond to the task space velocity inputs corresponding to the user and autonomous system respectively. Since full control of the platform is exerted, both inputs are 5-DOF, essentially the derivative of p_q from Eqn. 2. There is no secondary task, since the primary task is a full 5-DOF input and there is no inherent redundancy.

An expected use case for this type of control for a rover would be when either a task is very delicate, difficult, and not very dynamic, so the user has full control of the platform or very dull, so full autonomous control can be given to the rover.

Coordinated control involves easing the human operators task, by allowing them to control the robot in a lower dimensionality. The robot takes care of mapping the lower dimension input to the full system. An example of this can be envisioned where the camera the operator is using for situational awareness and driving is mounted on the first link of the arm. The operator may want to keep the camera view steady as the robot is moving, then the operator is providing a 3-DOF input consisting of the desired translation in the plane and the desired camera orientation. The robot resolves the redundancy between the rotation of the arm and base to achieve the desired platform motion while holding the camera view. With this is mind, the primary and secondary tasks are described by $\dot{p}_p$ and $\dot{p}_s$ as follows,

$$\dot{p}_p = \begin{bmatrix} \dot{x}_v & \dot{y}_v & \dot{\theta}_1 \end{bmatrix} \tag{11}$$
$$\dot{p}_s = \begin{bmatrix} \dot{\theta}_v & \dot{\theta}_2 \end{bmatrix} \tag{12}$$

In Eqn. 11 the user input provides the primary task as the

desired position of the robot and desired direction to point the camera mounted to the arm. The secondary task, Eqn. 12, takes care of minimizing the rotation in the base since it is easier to pan the arm rather than the whole base and also keeping the end-effector off the ground.

Collaborative control is very similar to the coordinated control above, but the user is given a lower dimensionality of inputs that is a direct subset of the joint space of the robot as opposed to an arbitrarily defined linear combination. In this case, the user is given full control of the robot base position, orientation, and arm orientation. The robot controls the arm elevation to keep the arm above the ground. This mode would be very useful for alignment of a sample before attempting to collect the sample with the end-effector, a task we identified as a good candidate for additional automation in [12]. The primary and secondary tasks are described by,

$$\dot{p}_p = \begin{bmatrix} \dot{x}_v & \dot{y}_v & \dot{\theta}_v & \dot{\theta}_1 \end{bmatrix} \tag{13}$$

$$\dot{p}_s = \begin{bmatrix} \dot{\theta}_2 \end{bmatrix} \tag{14}$$

Eqn. 13 assigns the primary task for the robot to follow the the commands of the human operator for positioning the robot base and orientation of the arm. Eqn. 14 assigns the secondary task to the robot to keep the end-effector above the ground so the user can slowly "bump" the platform or arm into a better position to pickup the sample.

Virtual constraint control involves defining a constraint, usually in task space, that limits the input the user can have on the system. The user has full control of the robot as long as the constraint is fulfilled. For example, assume that a ground-facing sensor is mounted on the arm and a survey needs to be conducted in front of the robot in a series of lines perpendicular to the direction of travel for the robot. The sensor height needs to be modulated above the ground to either follow terrain or conduct the survey at different heights. Because the survey lines need to be straight, but the arm is a fixed length, we can utilize the redundancy to follow the survey line with the end-effector while allowing the user to guide the location of the line and sampling point on that line without worrying about following the line itself. The primary and secondary tasks are described by,

$$\dot{p}_p = \begin{bmatrix} \dot{x}_v & \dot{y}_v & \dot{\theta}_1 \end{bmatrix} \tag{15}$$

$$\dot{p}_s = \begin{bmatrix} \dot{\theta}_v & \dot{\theta}_2 \end{bmatrix} \tag{16}$$

Eqn. 15 defines the primary task of keeping the end-effector over the line by moving the base within the plane, and rotating the first link of the arm so the sensor is over the scan line. This primary task is driven by the user input which is a desired location of the line in front of the robot, and the desired location on the scan line that needs to be surveyed. Eqn. 16 sets the secondary task to minimize the rotation of the base (implemented by the controller minimizing vibrations so the sensor is relatively undisturbed during scanning) and also keep the sensor at the desired height. A sensor that might require this type of operation would be a ground penetrating radar (GPR) used to create a 3-D model of the subsurface strata.

Blended shared control is the most collaborative modality where the human input and autonomous input are both controlling the aspects of the robot simultaneously. Implementation of blended shared control is difficult because the two inputs may be conflicting, and such cases must be easily detected. This type of control would be useful when approaching a region of interest that the rover has autonomously detected as something that needs to be further explored. The region may not be well defined, so a little assistance from the human operator would help guide the rover to the right place. In this case, both the human and robot drive towards the region of interest. If the two inputs are essentially going towards the same goal, the robot constantly receives positive reinforcement that it is on the correct path. On the other hand if the goal of the two inputs diverges too much, the robot can realize that there is a misunderstanding between where the human operator wants to explore and where the robot thinks it should be going. The primary task is a composition of both human and autonomous input described by,

$$\begin{aligned} \dot{p}_p &= \frac{\dot{p}_u + \dot{p}_a}{2} \\ &= \begin{bmatrix} \frac{\dot{x}_u + \dot{x}_a}{2} & \frac{\dot{y}_u + \dot{y}_a}{2} & \frac{\dot{\theta}_{v,u} + \dot{\theta}_{v,a}}{2} \end{bmatrix} \end{aligned} \tag{17}$$

$$\dot{p}_s = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 \end{bmatrix} \tag{18}$$

In Eqn. 17 both the user input and robot behavior are providing the translation motion and heading the robot should be facing. The two inputs are averaged, and if the user and robot are heading in the same general region, the two vectors and their average should be very similar and thus the robot will head in that direction constantly reassured by the operator input that it is behaving correctly. Eqn. 18 takes care of keeping the arm out of the way while navigating.

An antagonism metric needs to be defined so that if two inputs differ too much, then the system can recognize that it needs to stop and re-evaluate what it thinks it should be doing. There are many possible ways to define such a metric, but one simple way is using the vector dot product to judge how far apart the vectors are in both heading and magnitude. Eqn. 19 calculates the dot product of the two input vectors and normalizes the results so if it approaches zero, below a certain threshold, the system decides the two inputs are diverging from the same goal. The min/max ensures that the metric approaches zero no matter which vector is larger.

$$\beta = \frac{\dot{p}_u \cdot \dot{p}_a}{max(\dot{p}_u \cdot \dot{p}_u, \dot{p}_a \cdot \dot{p}_a)} = min\left(\frac{|\dot{p}_u|}{|\dot{p}_a|}, \frac{|\dot{p}_a|}{|\dot{p}_u|}\right) cos(\gamma) \tag{19}$$

Of course, weightings or other more complicated metrics could be implemented depending on what characteristics are important, but these would be defined on an task-centric basis.

5. Future Work and Conclusion

Future work needs to focus on expanding the control modalities, validating the theory and methods in simulation, and ultimately proving the applicability of the architecture on

real rovers. We have demonstrated the architecture in the applications that inspired its development, disaster robots and assistive robots, but space exploration robotics is an entirely different domain with significantly different challenges. Significant work needs to be expended to create a realistic simulation using models that capture the communication, mobility, and control challenges in conducting operations in space environments. The different control modalities and their associated use cases also need to be evaluated in simulation with respect to their applicability and whether they provide improvement over the current scripted teleoperation used to control space rovers. Ultimately, the architecture needs to be implemented and tested in an Earth analog test environment with conditions expected in space.

In addition, challenges remain in fully developing the theory and science behind an intuitive and functional shared control system. Traditional performance metrics are not necessarily applicable in shared control systems, so new metrics that are specific to shared control can be discussed and analyzed using the architecture. Integrating the work from [13] into the architecture would provide an excellent path. with human-in-the-loop systems, humans may be able to handle more tracking error subconsciously in exchange for better intuitive "feel" of a given controller. Since human-in-the-loop control involves close interaction between humans and robots, a trust metric would be a prime candidate [14].

We have presented a shared control architecture to enable the integration of operator input as constraints on the kinematic model of human-in-the-loop control systems for space exploration. We modified an existing architecture, originally developed for assistive and disaster response robotics, for potential use in a space exploration scenario. Several examples demonstrating the usefulness of different elements in space settings were outlined. A method for integrating human input with the autonomous behavior of the system using redundancy resolution taking into account primary and secondary tasks was presented. The result is that the operator input becomes a constraint on the kinematic model of the system, and traditional linear algebra tools can be used to determine the appropriate input to the system in order to complete a given task in a semi-autonomous manner.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1135854, 1264588, and 1355623. Additionally, the material is based upon work supported by the Defense Advanced Research Project Agency, DARPA Robotics Challenge Program under Contract No. HR0011-14-C-0011.

REFERENCES

[1] J. P. Grotzinger, J. Crisp, A. R. Vasavada, R. C. Anderson, C. J. Baker, R. Barry, D. F. Blake, P. Conrad, K. S. Edgett, B. Ferdowski *et al.*, "Mars science laboratory mission and science investigation," *Space science reviews*, vol. 170, no. 1-4, pp. 5–56, 2012.

[2] S. Srinivasa, D. Ferguson, C. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. Weghe, "Herb: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, pp. 5–20, 2010. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9160-9

[3] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *Control Systems Technology, IEEE Transactions on*, vol. 17, no. 5, pp. 1105–1118, Sept 2009.

[4] V. Dimitrov, M. DeDonato, A. Panzica, S. Zutshi, M. Wills, and T. Padir, "Hierarchical navigation architecture and robotic arm controller for a sample return rover," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, Oct 2013, pp. 4476–4481.

[5] A. Bejczy, W. Kim, and S. Venema, "The phantom robot: predictive displays for teleoperation with time delay," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, May 1990, pp. 546–551 vol.1.

[6] C. Lundberg and H. Christensen, "Evaluation of mapping with a tele-operated robot with video feedback," in *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, Sept 2006, pp. 164–170.

[7] M. W. Carey, E. M. Kurz, J. D. Matte, T. D. Perrault, and T. Padir, "Novel eod robot design with dexterous gripper and intuitive teleoperation," in *World Automation Congress (WAC), 2012*, June 2012, pp. 1–6.

[8] V. Dimitrov and T. Padir, "A shared control architecture for human-in-the-loop robotics applications," in *RO-MAN, 2014 IEEE*, Aug 2014.

[9] T. Padir, "Kinematic redundancy resolution for two cooperating underwater vehicles with on-board manipulators," in *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, vol. 4, Oct 2005, pp. 3137–3142 Vol. 4.

[10] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *Int. J. Rob. Res.*, vol. 6, no. 2, pp. 3–15, Jul. 1987. [Online]. Available: http://dx.doi.org/10.1177/027836498700600201

[11] A. Enes and W. Book, "Blended shared control of zermelo's navigation problem," in *American Control Conference (ACC), 2010*, June 2010, pp. 4307–4312.

[12] V. Dimitrov and T. Padir, "A comparative study of teleoperated and autonomous task completion for sample return rover missions," in *Aerospace Conference, 2014 IEEE*, March 2014, pp. 1–6.

[13] J. Saleh and F. Karray, "Towards generalized performance metrics for human-robot interaction," in *Autonomous and Intelligent Systems (AIS), 2010 International Conference on*, June 2010, pp. 1–6.

[14] M. Desai, M. Medvedev, M. Vazquez, S. McSheehy, S. Gadea-Omelchenko, C. Bruggeman, A. Steinfeld, and H. Yanco, "Effects of changing reliability on trust of robot systems," in *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*, March 2012, pp. 73–80.

BIOGRAPHY

Velin Dimitrov *is a Ph.D. Candidate in the Electrical and Computer Engineering department at Northeastern University. Velin received his Bachelors of Science in Electrical and Computer Engineering from Franklin W. Olin College of Engineering in Needham, MA, and his Masters of Science in Robotics Engineering from Worcester Polytechnic Institute. As part of his work in the Robotics and Intelligent Vehicle Research (RIVeR) Laboratory, Velin has been involved in NSF sponsored research on developing a semiautonomous wheelchair, DARPA funded research working with the ATLAS robot for the DARPA Robotics Challenge, and participated in two NASA RASCAL Robo-Ops competitions and two NASA Sample Return Robot Centennial Challenges. Velins areas of interest include human-in-the-loop shared control of robots for of space exploration, disaster response, and assistive applications.*

Taşkın Padır *is an Associate Professor of Electrical and Computer Engineering at Northeastern University. He received his Ph.D and M.S. degrees in electrical and computer engineering from Purdue University. He holds a B.S in electrical and electronics engineering from the Middle East Technical University in Turkey. Dr. Padir has significant experience with the design, development and control of robotic systems and intelligent vehicles. He is the director of the Robotics and Intelligent Vehicles Research Laboratory (RIVeR Lab) at Northeastern. His research interests include human-in-the-loop robotic systems, design of robot control interfaces, cooperating robots, control of redundant robot systems, control of ground vehicles, navigation, path planning, and mapping for autonomous robots. His projects have been sponsored by NSF, DARPA, NASA, AFRL, Draper Laboratory and many industry partners including AgCo, The MathWorks, Solidworks, and National Instruments. Moreover, he was the 2013 Joseph S. Satin Distinguished Fellow in WPI's Electrical Engineering, and he received the Inaugural Rho Beta Epsilon Award for Excellence in Robotics Education in 2010 and 2011 Romeo L. Moruzzi Young Faculty Award for Innovation in Undergraduate Education for his contributions to WPI's unique undergraduate program in robotics engineering. He is currently the PI of two NSF sponsored grants on the design and control of assistive robotics systems and was a co-PI on WPI's DARPA Robotics Challenge Track C Team which took the 2nd place in the Virtual Robotics Challenge in June 2013, 7th place in the DRC Trials, and advanced to DRC Finals to be held in June 2015.*