Human-in-the-Loop: MPC for Shared Control of a Quadruped Rescue Robot

Rahul Chipalkatty, Hannes Daepp, Magnus Egerstedt, and Wayne Book
chipalkatty@gatech.edu, hdaepp@gatech.edu, magnus@ece.gatech.edu, wayne.book@me.gatech.edu
Georgia Institute of Technology, Atlanta, GA 30332, USA

***Abstract*— In this paper, we present a control theoretic formulation for composing human control inputs with an automatic controller for shared control of a quadruped rescue robot. The formulation utilizes a model predictive controller to guide human controlled leg positions to satisfy state constraints that correspond to static stability for the robot. A hybrid control architecture that incorporates the model predictive controller is developed to implement a gait that guarantees stable foot-placements for the robot. The algorithm is applied to a simulation of a quadruped rescue robot with human input provided through haptic joysticks.**

I. INTRODUCTION

Urban Search and Rescue (USAR) robotics is an important domain for Human-Robot Interaction (HRI) research, as it requires the high-level skills only human operators are currently capable of providing while trying to take advantage of the safety and efficiency of automated tasks that state-of-the-art robotics affords. In the USAR scenario, humans are kept in the control loop, while being safely out of harm's way. However, integrating human control with automatic controllers or automated tasks is an open research area. Therefore, we propose to utilize a set of control theoretic tools (as in [1]) to compose human-controlled front foot-placement commands with an automatic controller that prevents unstable foot-placement for a quadruped rescue robot.

The Compact Rescue Robot (CRR) is a pneumatically-actuated tele-operated rescue robot being developed by the Intelligent Machine Dynamics Laboratory (IMDL) at Georgia Tech (Figure 1). The quadruped robot is a testbed of the Center for Compact and Efficient Fluid Power, which seeks to demonstrate the benefits of fluid power to engineering technologies. The CRR is controlled using an interface that allows the two front legs to be manipulated by a human operating two corresponding Phantom^{TM} haptic joysticks.

The operator can utilize the haptic feedback to aid in foot-placement in uneven terrain commonly found in disaster sites. However, in the previous configuration, the user received little to no feedback on whether or not these foot-placements would lead to an unstable leg configuration that would cause the robot to fall over. This lack of situational awareness for stability warrants an online controller that tries to both preserve the intent of the human operator and satisfy the stability constraints of the robot, i.e. by sending motor commands that minimize errors from the human command while still satisfying a constraint. In other words, human operators are effective in navigating the robot in rough terrain, but are poor at preventing the robot from falling over.

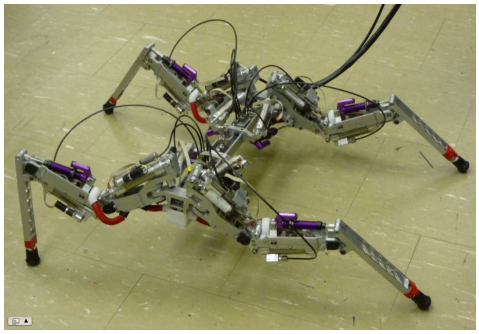

Fig. 1. Quadruped Rescue Robot

An effective shared control framework would combine the strengths of both operator and autonomous stability control.

Researchers in the HRI field refer to the proposed control interaction between human and controller as mixed initiative interaction or human-in-the-loop (e.g., [2],[3]). It can also be referred to as shared control, as in [4] and [5], since both controllers (human and computer) act on the same dynamic system. One of the strengths of the control presented in [1] is the varying levels of autonomy that result from the interaction of human and automatic control. The authors in [6] highlight the advantages of having varying levels of autonomy for robots in such rescue situations. The authors argue that teleoperation may be required for rescue robots to navigate complicated and cluttered terrain, while some rescue operations may require a higher level of robot autonomy to compensate for a lack of human situational awareness, stemming from limited perception of the work environment.

Previous work on shared control frameworks include work on mobility aids, as in [5], where the algorithms are rule-based, and need to be experimentally tuned in order to smooth transitions between user control and automatic control. In [4], shared control for vehicle steering was examined using a motorized steering wheel and human driver, where the control relies on the human to provide physical force to impart the intended behavior on the system. In the proposed work, we invert this relationship in that the controller is guiding the human to reach a target set of leg positions defined by a state constraint. This allows the automatic controller to ensure that state constraints on the system are satisfied, while the human can direct the system to a solution, within the constraints that is suitable from a higher-level point-of-view.

Some past work in the areas of teleoperation and control of a quadruped robot in cluttered environments and rough terrain seek to provide greater situational awareness of the

work environment through the use of haptic feedback (e.g. [7],[8], [9]). Additionally, prior research has addressed robot stability for legged robots in environments with uneven terrain, as in [10], [11], [12], [13], and [14].

Specifically, in this paper, we utilize a control framework, as in [1], to implement a model predictive optimal controller that applies user foot-placement commands as closely to what the human intended as possible while still maintaining static stability for the next leg placement as well as the rest of the proposed robot gait. The front leg controller is updated with the latest leg positions and human input commands and is composed with human commands so that the stability constraint is not violated at foot touch-down, i.e. the front foot is never placed on the floor such that the center of mass of the robot is outside the polygon created by that leg and the hind legs. This allows for the pick-up and placement of the other front leg. This is demonstrated on a simulation testbed of a quadruped rescue robot.

II. Problem Formulation

In the following section, we model how the robot's feet must be placed in order to avoid losing static stability and how the proposed robot gait and control framework is constructed to accomodate human commands and automatic movements. A planar robot model is proposed where all leg and center of mass positions in $\mathbb{R}^3$ are projected onto the ground plane ($\mathbb{R}^2$) as in Figure 2. This is a typical formulation in the quadrupedal robot stability literature (e.g. [10], [15], and [16]). All positions are given in reference to a global coordinate frame.

As discussed in [10], in order to pick up and place a leg, the center of mass must lie in the triangle formed by the other three legs (commonly referred to as the stability polygon). If this is the case, then, for example, the left leg must be picked up and placed in a position such that right leg can be picked up and placed. This requires that the first leg be placed in such a way that it forms a triange with the two remaining legs that contains the center of mass. The stability polygons for both left and right front leg are shown respectively as the dashed and dash-dotted triangles in Figure 2.

In this paper, we represent the set of leg placements that lead to the center of mass being in these stability polygons as the stability cones for the front and back legs as shown in Figure 2. We will first address robot stability, then will discuss the gait proposed in Section II-B and the subsequent proposed front leg control.

A. Stability Cone

Referring to Figure 2, in order to guarantee a stable right leg lift and place, the front left leg should be placed in the half plane defined by the line passing through the center of mass and back right leg. When the left leg is placed in this half-plane, then the left leg necessarily creates a stability polygon with the back legs such that the center of mass lies within this polygon. Hence, the robot is statically stable when the right leg is lifted. Similarly, the front right leg should be placed in the half plane defined by the line passing

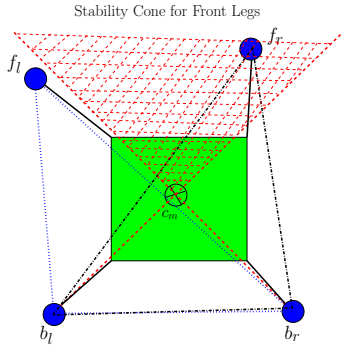

Fig. 2. The Front Leg Stability Cone is shown in the dotted hatching, while the stability polygon for the right front leg is shown as the dotted triangle and the stability polygon for the left leg is shown as the dash-dotted triangle.

through the center of mass and the back left leg. Together, the intersection of these two half-planes define the *stability cone*. If the front left leg is not placed in the stability cone, then the subsequent front right leg placement will result in static instability and the robot will fall.

An analytic expression is derived for this cone using lines orthogonal to the line connecting the center of mass to the back legs. The following notation is used: f_l is the position of the front left leg, f_r is the position of the front right leg, b_r is the position of the back right leg, b_l is the position of the back left leg, and c_m is the position of the center of mass. All positions are in $\mathbb{R}^2$. R_θ is the planar rotation matrix evaluated at angle θ. The half-plane defining where the front left leg can be placed is given by

$$R_{-\frac{\pi}{2}}(b_r - c_m)^T(x - c_m) \geq 0$$

where x represents all front left leg positions that allow for the stable lifting of the front right leg. Similarly, the half-plane defining where the front right leg can be placed is given by

$$R_{\frac{\pi}{2}}(b_l - c_m)^T(x - c_m) \geq 0$$

where x represents all front right leg positions that lead to stable lifting of the front left leg. As such, the intersection of these two half-planes define the stability cone giving $C_f x \geq e_f$, where

$$C_f = \begin{bmatrix} R_{-\frac{\pi}{2}}(b_r - c_m)^T \\ R_{\frac{\pi}{2}}(b_l - c_m)^T \end{bmatrix}, \; e_f = \begin{bmatrix} R_{-\frac{\pi}{2}}(b_r - c_m)^T c_m \\ R_{\frac{\pi}{2}}(b_l - c_m)^T c_m \end{bmatrix}$$

for all x representing stable front leg positions. In the same vein, the stability cone for the back legs is given by $C_b x \geq e_b$ with

$$C_b = \begin{bmatrix} R_{\frac{\pi}{2}}(f_r - c_m)^T \\ R_{-\frac{\pi}{2}}(f_l - c_m)^T \end{bmatrix}, e_b = \begin{bmatrix} R_{\frac{\pi}{2}}(f_r - c_m)^T c_m \\ R_{-\frac{\pi}{2}}(f_l - c_m)^T c_m \end{bmatrix}$$

for all x representing stable back leg positions. Now that a formal definition of the stability cone is given for the robot, we can address details of the proposed gait.

B. Gait

The gait sequence will be Move 1: front left leg, Move 2: front right leg, Move 3: center of mass shift forward, Move 4: back right leg, Move 5: back left leg, and Move 6: center of mass shift backward; then, the cycle repeats. The particular gait proposed in this work allows for the placement of the

front two legs in succession without a center of mass shift. This is desirable in that one set of position constraints can be used for both front leg movements, which are purely a function of the center of mass and the back legs, and not of the position of the other front leg.

Move 3 is necessary to guarantee that the back right leg can be picked up, since the front right leg is only placed in the stability cone that guarantees stability for left leg placements. This shift also serves the purpose of widening the stability cone for the back legs by moving the center of mass closer to the front two legs as well as dictating the direction the entire robot will move in.

Next, Moves 4 and 5 will place the back two legs in the back leg stability cone, and this portion of the control is fully automated so no human input is needed. The back legs are moved in the same general direction that the front two legs are moved in and are required to be in the back leg stability cone. Lastly, Move 6 moves the center of mass back towards the back legs to widen the stability cone for the front legs and guarantees that the front left leg can be moved.

It is important to note that the proposed gait is a departure from the commonly accepted gait (as in [17]), where the back leg placements start the gait. However, in this paper, the human input guides not only the front legs but the general robot motion, so it is required that the front leg placements initiate the proposed gait. After formally defining the stability cone and gaits, the next section details how each of these leg movements are implemented in a control framework.

C. Dynamic System and Hybrid Control Framework

Human input to the front legs is provided by a change in position vector. As such, the following are the linear discrete dynamics for the front legs,

$$f_l^{k+1} = f_l^k + u_l^k,$$
$$f_r^{k+1} = f_r^k + u_r^k,$$

where the superscript k refers to the current discrete time and $k+1$ is the subsequent time step.

The discrete dynamics describing the back legs and center of mass are formulated differently. This is because low-level controllers achieve the desired positions via displacement commands, $u_{b_l}^k$ for the back left leg,$u_{b_r}^k$ for the back right leg, and u_c^k for the center of mass. The low-level functions are discussed in Sections III and IV. The back legs and center of mass evolve in discrete time as

$$b_l^{k+1} = g_l(b_l^k, u_{b_l}^k),$$
$$b_r^{k+1} = g_r(b_r^k, u_{b_r}^k),$$
$$c_m^{k+1} = g_c(c_m^k, u_c^k).$$

Collecting the dynamics, let $X^k = (f_l^k, f_r^k, b_l^k, b_r^k, c_m^k)$ and $U^k = (u_l^k, u_r^k, u_{b_l}^k, u_{b_r}^k, u_c^k)$ be column vectors which lead to the dynamics of the system,

$$X^{k+1} = G(X^k, U^k).$$

Let the subscript l in U_l^k denote the vector U^k where every control element is zero but u_l^k, e.g. $U_l^k = (u_l^k, 0, 0, 0, 0)$.

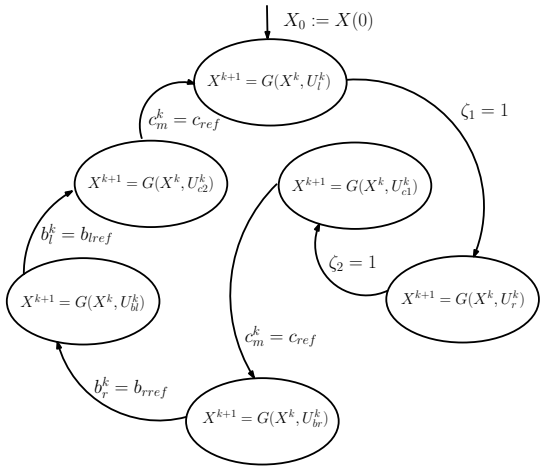

Fig. 3. Quadruped Rescue Robot Hybrid Automoton

This notation follows for the vectors U_r^k, U_{bl}^k, U_{c1}^k, and U_{c2}^k. since the system is decoupled and only one leg or center of mass is moved at a time.

Since each leg and the center of mass is controlled individually, we describe the system as a hybrid system modeled with the hybrid automaton in Figure 3. In the figure, $X(0)$ denotes the initial leg and center of mass positions. The guards for the front legs are simple flags set by the user to indicate completion of a particular leg placement. The guards for the other movements are set when the leg reaches the desired position calclulated by the automatic control.

III. MPC-BASED SHARED CONTROL: FRONT LEGS

With the dynamic model and control structure defined, we formulate the control laws needed to both incorporate human commands for the front legs as well as guaranteeing static stability for the rest of the gait. Let $\{v^k\} = \{v^k, \ldots, v^{k+N-1}\}$ be a sequence of predicted human inputs that drive one of the front legs for N discrete time steps into the future (N is called the control horizon) and let $\{u^k\} = \{u^k, \ldots, u^{k+N-1}\}$ denote the applied control sequence for one of the front legs, where $u^k = u_l^k$ or $u^k = u_r^k$ and $x^k = f_l^k$ or $x^k = f_r^k$, depending on which leg is being placed (i.e. Move 1 or Move 2). The predicted human input is a linearly extrapolated sequence starting with u^k and with slope $u^k - u^{k-1}$ as in [1].

The following controller minimizes deviations from the predicted human input sequence while satisfing the constraint that the leg position be in the stability cone at the end of the control horizon. As such, the model predictive optimal controller solves the following optimal control problem, $\mathcal{P}_N$, at each discrete time instant k,

$\mathcal{P}_N$:

$$\min_{\{u^k\}} \sum_{i=k}^{k+N-1} (u^i - v^i)^T (u^i - v^i),$$

such that

$$\begin{aligned} x^{k+1} &= Ax^k + Bu^k, \\ x^{k+N} &\in \mathbb{X}_f = \{x \mid C_f x \geq e_f\}. \end{aligned} \tag{1}$$

where A and B are equal to the identity matrix in $\mathbb{R}^2$ for the front legs.

The control, u^k, applied to (1) is the first element in the sequence $\{u^k\}$. Due to the inequality constraint, this optimal control problem will be solved numerically at each time instant. This is where the control formulation presented in this paper differs from the control problem solved in [1], where an analytic solution was given for an equality constraint. In [1], it is shown that this particular model predictive control will result in the state (i.e. leg position) asymptotically converging to the constraint set, i.e. the stability cone. The control ceases when the user sets a completion flag, thus setting $\zeta_1 = 1$ when the left leg control is completed, and $\zeta_2 = 1$ when the right leg control is completed.

A low-level control function is now needed to implement the commanded front leg displacement as leg joint commands that result in the desired motion on the robot leg. This motion of the front legs is achieved by mapping the end effector position of the Phantom joysticks to local leg workspaces on the shoulder, and then mapping these positions to desired robot joint angles. With the proposed controller, the user's commands are adjusted, resulting in a sense of haptic guidance. Together with the low-level control functions, we now have a controller for U_r^k and U_l^k as seen in the hybrid automaton diagram in Figure 3.

IV. Center of Mass and Back Leg Control

The first center of mass shift moves the center of mass towards the front two legs as well as guaranteeing stability for a back right foot placement. Hence, we define a constraint restricting the center of mass to lie in the stability polygon formed by the front two legs and the back left leg. The general direction of robot motion is defined as the vector from the center of mass to the midpoint of the line segment connecting the front two legs. In this way, the human operator not only has input on where the front legs are placed, but also on the general direction that the robot will move in. A constrained quadratic program is then solved that provides a change of position command that tries to move the center of mass in the general direction of robot motion while guaranteeing stability for a back right leg lift and place.

The intent is to then have the low-level center of mass control function shift the robot's center in the commanded direction, while keeping the feet fixed. As is standard practice for parallel robots, the robot is interpreted as four serial robots anchored to the ground by ball-and-socket joints that are mutually linked by a platform. New joint angles are calculated based on incremental motions of this platform with fixed end effector positions. These joint angles are commanded simultaneously to each leg at every update.

Similarly, we solve quadratic programs to move the back legs in the general direction of robot motion, but we restrict it to the back leg stability cone. To execute the second center of mass shift, the center of mass is moved back towards the back legs to widen the cone for the front legs and guarantee a stable leg configuration to move the front left leg.

The desired displacement of the back leg position is fed to the back leg low-level control function, where a third-order curve is fit to provide the necessary trajectory. This trajectory is, then, mapped to leg joint angles, and commanded to the robot while the other legs are held steady. Together with the low-level control functions discussed in this section, these leg controllers complete the control framework required for the proposed gait.

Fig. 4. Operator Interface and CRR Simulation

V. Testbed Implementation

The proposed controller is implemented on a physical setup consisting of an operator interface and corresponding robot. Because the physical robot is currently in development, a dynamic simulation (Figure 4) is used in the robot's stead.

A. Hardware

The operator interface, depicted in Figure 4, consists of two three degree-of-freedom haptic joysticks that use admittance control to relate the operator's desired motions to the actual ones on the robot. Audio-visual feedback is provided through a headset, ensuring that the operator has extensive tele-presence. Each of the end effectors of the Phantom joysticks maps to an end effector of a robot leg and a set of switches is used to turn on the controller and switch between gait states. The joystick software is programmed in C and provides output that is sent via UDP to the robot, which uses a 1.4 GHz PC104 computer running xPC Target/Simulink coupled with a 2.9 GHz host computer.

B. Robot Dynamic Simulation

The software used is a dynamics library known as SrLib, developed by Seoul National University. This simulation accepts the same desired leg joint angles that would otherwise be sent to position controllers on the actual robot, and returns actual joint angles, position, and orientation, exactly as provided on the actual robot.

C. Adjustments and Limitations

The described hardware and software configuration enables a proof-of-concept and a demonstration of the feasibility of the user interface. However, there are a few limitations.

First, while the simulation is able to simulate near ideal conditions, some slippage still occurs. To compensate for this effect, the desired center of mass is updated during shifting, which is done by changing the line along which the desired center of mass is projected as the feet slide.

Second, the center of mass shifts and rear leg motions use path trajectories that are designed based on general

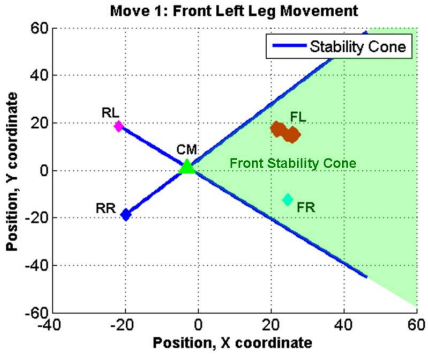

Fig. 5. Front left leg moving freely within the stability cone limits as commanded by the operator. FL = Front Left, FR = Front Right, RL = Rear Left, RR = Rear Right, CM = Center of Mass.

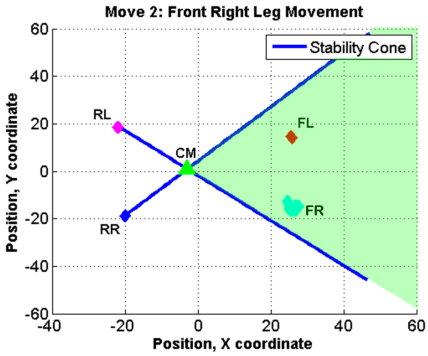

Fig. 6. Robot in a statically stable position while the front right leg is moved freely within the stability cone limits as commanded by the operator.

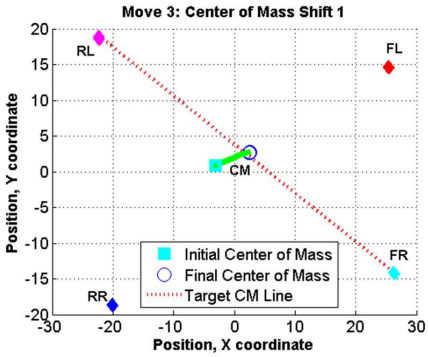

Fig. 7. First center of mass shift ensures that the right leg can moved while maintaing static stability.

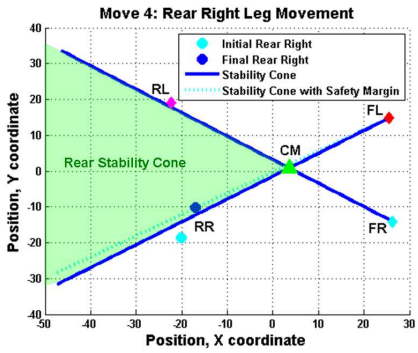

Fig. 8. Rear right leg motion results in a statically stable configuration that allows the left leg to be moved.

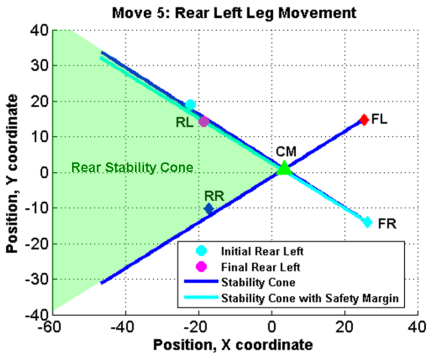

Fig. 9. Rear left leg is moved inside the cone.

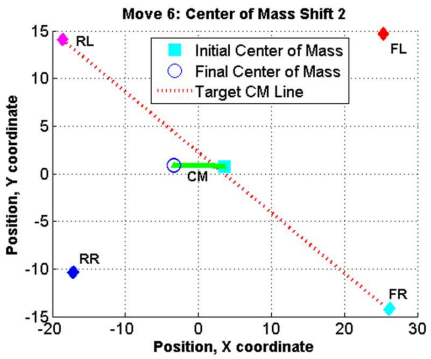

Fig. 10. Second center of mass shift widens the cone for freer motion constraints on the front legs.

feasibility, and have no measure for dealing with joint angle limits or other predefined obstacles. This could be improved for field implementation by using planners to determine the paths taken between the start and end positions.

VI. Experimental Results

A complete gait cycle, starting from a statically stable position, was run, and the results were visually documented using the top-down labelled views shown in Figures 6 - 11. The goal here was to demonstrate that the gait developed in this paper can be successfully implemented, ensuring that the user's motion was appropriately constrained and that the robot always maintained a statically stable position, while also contributing towards forward motion. The proposed front leg controller was implemented with a control horizon of $N = 5$ for a 0.002 second sampling rate.

Figure 5 depicts the initial position, as well as Move 1: the user's motion of the front left leg is constrained by the front stability cone, which is represented by the shaded green area within the blue lines. Physically, the user is able to feel these limits through the use of haptic joysticks in the form of moving into a soft wall.

After placing the front left leg, the user cycles to Move 2 and places the front right leg. Figure 6 shows that placement of this leg is again constrained by the stability cone.

Move 3, shown in Figure 7, is the first center of mass shift. Here, the center of mass (labelled in its final position) is shown to be effectively shifted forward to a desired center of mass with a safety factor to ensure that the robot is not merely marginally stable.

Following the first center of mass shift, the rear legs are moved autonomously from their original point to a

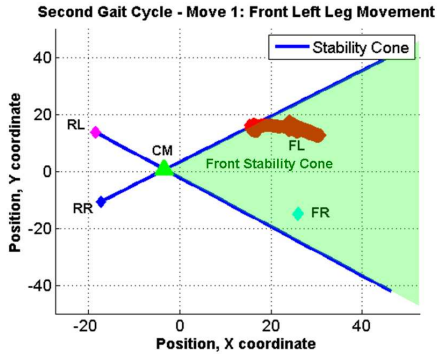

Fig. 11. Second gait cycle, the front left leg can again be placed successfully be the user within the range defined by the cone.

desired end location in Moves 4 and 5. Because of physical configuration constraints, extra measures were introduced in the rear right leg shift to ensure that the required motions would be made kinematically feasible. Specifically, instead of simply moving the leg from it's initial to final position, it was moved to a final 'safe' position, defined by the mechanical configuration of the robot, and then shifted the rest of the way to ensure that the final position ended inside the cone. While these extra measures are not ideal, they are an artifact resulting from reduced flexibility when working with an existing robot. Rather than demonstrating the entire motion, Figure 8 shows just the final position of the rear right leg following the right leg placement, which demonstrates that the leg is safely inside the cone. Here, the dashed cyan line again depicts the usage of a safety factor to ensure that the robot is not merely marginally stable.

The results of the rear left leg motion, Move 5, are shown in Figure 9. The left leg has successfully reached a statically stable final position along the dashed cyan line, which again represents the desired point within the cone including a given safety margin.

Move 6 is shown in Figure 10. This second shift is intended to widen the cone for the front legs and ensure continued forward motion. It can also be observed that the forward and rear legs have advanced in the general robot direction; therefore, forward locomotion is achieved.

Finally, Figure 11 shows the start of the next gait cycle. The user is again constrained by the stability cone, and is free to continue moving forward.

Therefore, these results show that the human operator is able to carry out the higher-level task of placing the front legs in locations they deem appropriate (i.e in cluttered environments, the leg can be placed on relatively flat surfaces), while the controller and automatic gait ensure the lower-level task of ensuring static stability is completed.

VII. Conclusions

In this work, we present a model predictive optimal controller that incorporates human input commands to control the front legs of a quadruped rescue robot as well as perform automated back leg and center of mass movements to ensure static stability. The controller guides the user to place each of the front legs in a way that ensures the next front leg movement or any subsequent leg and center of mass movements will result in static stability for the robot. The viability of the controller was demonstrated on a hardware-in-the-loop simulation with real human input.

References

[1] R. Chipalkatty and M. Egerstedt, "Human-in-the-loop: Terminal constraint receding horizon control with human inputs," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2712–2717.

[2] M. A. Goodrich and A. C. Schultz, "Human robot interaction: A survey," *Foundations and Trends in HumanComputer Interaction*, vol. 1: No 3, pp. 203–275, 2007.

[3] J. E. Allen, C. I. Guinn, and E. Horvtz, "Mixed-initiative interaction," *IEEE Intelligent Systems and their Applications*, vol. 14, no. 5, pp. 14–23, 1999.

[4] P. Griffiths and R. B. Gillespie, "Shared control between human and machine: haptic display of automation during manual control of vehicle heading," in *Proc. 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems HAPTICS '04*, 27–28 March 2004, pp. 358–366.

[5] G. Wasson, J. Gunderson, S. Graves, and R. Felder, "Effective shared control in cooperative mobility aids," in *In Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society Conference*. AAAI Press, 2001, pp. 509–513.

[6] I. R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human-robot teaming for search and rescue," *Pervasive Computing, IEEE*, vol. 4, no. 1, pp. 72–79, 2005.

[7] S. A. A. Moosavian, A. Kalantari, H. Semsarilar, E. Aboosaeedan, and E. Mihankhah, "Resquake: A tele-operative rescue robot," *Journal of Mechanical Design*, vol. 131, no. 8, p. 081005, 2009.

[8] S.-G. Hong, B. S. Kim, S. Kim, and J.-J. Lee, "Artificial force reflection control for teleoperated mobile robots," *Mechatronics*, vol. 8, no. 6, pp. 707 – 717, 1998.

[9] Q.-J. Huang and K. Nonami, "Humanitarian mine detecting six-legged walking robot and hybrid neuro walking control with position/force control," *Mechatronics*, vol. 13, no. 8-9, pp. 773 – 790, 2003.

[10] J. Estremera and P. de Santos, "Generating continuous free crab gaits for quadruped robots on irregular terrain," *Robotics, IEEE Transactions on*, vol. 21, no. 6, pp. 1067 – 1076, Dec. 2005.

[11] P. De Santos, J. Galvez, J. Estremera, and E. Garcia, "Sil04: a true walking robot for the comparative study of walking machine techniques," *Robotics Automation Magazine, IEEE*, vol. 10, no. 4, pp. 23 – 32, Dec. 2003.

[12] E. Celaya and J. Porta, "A control structure for the locomotion of a legged robot on difficult terrain," *Robotics & Automation Magazine, IEEE*, vol. 5, no. 2, pp. 43–51, 2002.

[13] J. Cobano, J. Estremera, and P. G. de Santos, "Location of legged robots in outdoor environments," *Robotics and Autonomous Systems*, vol. 56, no. 9, pp. 751 – 761, 2008.

[14] D. Aarno and D. Kragic, "Motion intention recognition in robot assisted applications," *Robotics and Autonomous Systems*, vol. 56, no. 8, pp. 692 – 705, 2008.

[15] J. Estremera, E. Garcia, and P. Gonzalez De Santos, "A multi-modal and collaborative human–machine interface for a walking robot," *J. Intell. Robotics Syst.*, vol. 35, no. 4, pp. 397–425, 2002.

[16] P. G. De Santos, J. Estremera, E. Garcia, and M. Armada, "Including joint torques and power consumption in the stability margin of walking robots," *Auton. Robots*, vol. 18, no. 1, pp. 43–57, 2005.

[17] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng, "A control architecture for quadruped locomotion over rough terrain," in *IEEE International Conference on Robotics and Automation. The Half-Day Workshop on: Towards Autonomous Agriculture of Tomorrow, 19-23 May*, 2008, pp. 4814–21.