3D Multi-Contact Gait Design for Prosthesis Using FROST

Elin Samuelsson[1]
Mentor: Aaron D. Ames[2], Co-mentor: Rachel Gehlhar[3], Advisor: Jacob Reher[3]

***Abstract*— Walking with an energetically passive prosthetic leg constrains the user in many ways. Powered prostheses are under development and hold potential to give an amputee full mobility in various terrains. The goal of this project is to simulate stable gait trajectories for the powered prosthetic leg AMPRO3. It should walk as human-like as possible. This includes the multi-contact behavior of respectively putting pressure on the heel and toe throughout one step.**

The human lower body is modeled as a chain of links and joints and the gait as a hybrid system with continuous domains and discrete transitions in between. Using optimization to minimize energy while fulfilling specific constraints, appropriate movements for each joint are found. In simulation, a feedback controller then drives the joints to follow these desired paths, generating the actual trajectories. The MATLAB toolkit FROST, developed by the AMBER lab for this kind of problem, is used as the control development framework. A natural next step is to implement the simulated trajectories on the actual prosthesis and perform experiments. The code is also easy to modify to generate gaits for other walking behaviors, such as walking on slopes and stair walking.

I. Introduction

In the United States, there are approximately 222,000 people suffering from transfemoral (above the knee) amputation [1]. The current commercial market mainly offers energetically passive prostheses, which constrain the user by both increased metabolic cost and limited locomotion capabilities [2]. To address this, powered prosthesis are under development. They hold potential to enable an amputee to walk more efficiently and with a healthier gait.

In AMBER Lab, led by Aaron D. Ames, the prosthetic leg AMPRO3 is currently being built. An earlier version, AMPRO [3], is shown in Fig. 1.

In this project, human-like walking with AMPRO3 on a flat surface is simulated. The method can be divided into three steps: modeling, trajectory optimization and simulation. The model of an amputee wearing AMPRO3 is simply a system of links and joints. Due to the combination of continuous and discrete behavior, the gait can be seen as a hybrid control system. The steps are divided into several phases, where each phase has certain constraints that need to be fulfilled. An optimization method finds stable, symmetric joint trajectories that satisfy all constraints. In the simulation, the amputee-prosthesis model tries to follow those optimal paths, using a feedback controller. The resulting simulated walking trajectories can later be used as input to the real AMPRO3.

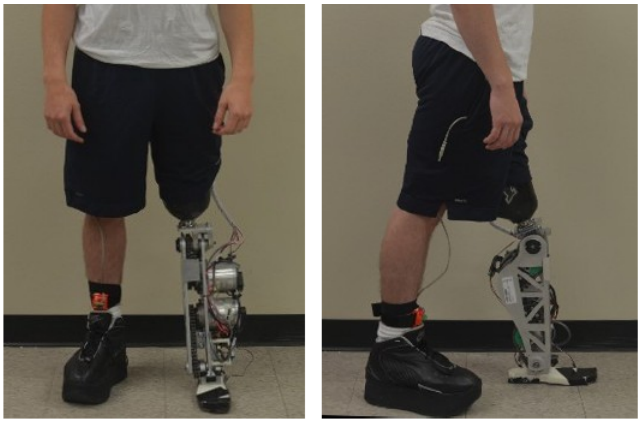

Fig. 1. Transfemoral test subject wearing an earlier version of the powered prosthetic device, named AMPRO [3].

The problem is set up in MATLAB, using the software FROST [4]. It stands for Fast Robot Simulation and Optimization Toolkit [4] and is custom-made for dynamical locomotion. Compared to old implementations, code written in FROST has a quicker runtime. Problems are also easier to modify. These advantages motivates the purpose of this project: to implement a human-prosthesis model in FROST and simulate human-like gaits.

The project is divided into generating the following types of gaits:

- **Single domain gait:** Only one foot is in contact with the ground at a time.
- **Flat-foot gait:** The model alternates between standing on one and two feet.
- **Multi-contact gait:** Pressure is put on heel and toe, respectively, as shown in Fig. 2 from [5].

As the complexity increases, we approach actual human gait. Rolling over the foot, from heel to toe, is crucial. It makes humans walk fluidly and energy efficiently, and may also help reduce knee osteoarthritis in transfemoral amputees [6].

II. Theory

The amputee-prosthesis system is treated as a bipedal robot. Its state is determined by a set of coordinates $q \in \mathcal{Q}$. For bipedal robots, there is one 6-dimensional base

[1]Elin Samuelsson participates in the Summer Undergraduate Research Fellowship (SURF) program, California Institute of Technology, Pasadena, CA 91125, USA. elsamu@kth.se

[2]Aaron D. Ames are with the Department of Mechanical and Civil Engineering and the Department of Computing + Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA. ames@caltech.edu

[3]Rachel Gehlhar and Jacob Reher are with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91125, USA. {rgehlhar,jreher}@caltech.edu

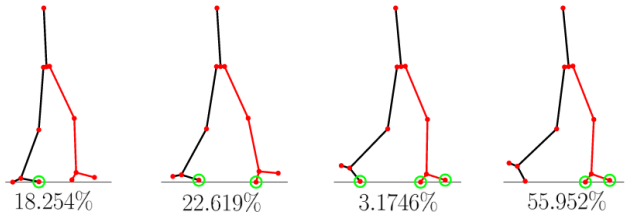

Fig. 2. Multi-contact gait that includes a foot roll, where pressure is put on the heel, then the toe. The percentages are the time spent in each domain, relative the whole step [5].

frame, defining the robot's position and rotation relative the surroundings. Further, the angles of the joints in the model are also used as coordinates. Therefore, the number of such joints + the six base coordinates equals the degrees of freedom n: $\mathcal{Q} \subset \mathbb{R}^{n\times 1}$. The state of the robot is defined as $x = (q, \dot{q}) \in X \subset \mathbb{R}^{2n\times 1}$ [7].

Bipedal locomotion has a cyclic behavior. Each step can be divided into continuous domains (e.g. when the leg swings forward) and discrete transitions (e.g. when the foot strikes the ground) in between. In a steady-state, these phases are ordered and periodic. [8] This motivates the use of a hybrid control system to model human-like gait:

$$\mathscr{HC} = (\Gamma, D, U, S, \Delta, FG), \tag{1}$$

where $\Gamma = (V, E)$ is a directed circle graph. Using the notation from [8]: $v \in V$ is an arbitrary vertex, denoting one of the continuous phases. v^+ is the subsequent vertex and $e = \{v \to v^+\} \in E$ is the discrete transition from v to v^+.

The following properties are defined uniquely for each vertex v: a domain D_v, a set of actuators U_v, a guard S_e, a reset map Δ_e and a control system (f_v, g_v).

A. Holonomic Constraints

The vertices are characterized by which parts of the feet are in contact with the ground. Any contact with the external environment generates some holonomic constraints $\eta_v(q)$. They require certain positions and rotations to be constant:

$$\eta_v \equiv constant, \tag{2}$$
$$J_v(q)\dot{q} = 0, \tag{3}$$

where $J_v(q) = \frac{\partial \eta_v}{\partial q}$ is the Jacobian matrix of η_v. The number of holonomic constraints is denoted n_v.

B. Continuous Dynamics

The mass and inertia properties of the links in the robot model are known. From them, the Lagrangian is set up as kinetic energy minus potential energy [6]:

$$\mathscr{L} = \frac{1}{2}\dot{q}M(q)\dot{q} - P(q). \tag{4}$$

Using the Euler-Lagrangian equations, the equation of motion for a given vertex v is written:

$$M(q)\ddot{q} + H(q, \dot{q}) = B_v u_v + J_v^T(q)F_v(q, \dot{q}, u_v), \tag{5}$$

where $F_v(q, \dot{q}, u_v) \in \mathbb{R}^{n_v\times 1}$ is a vector of contact wrenches, i.e. the constraint forces and/or moments corresponding to each of the holonomic constraints [8].

The holonomic constraints η_v are enforced by requiring their second order differentiation to equal zero [8]:

$$J_v(q)\ddot{q} + \dot{J}_v(q, \dot{q})\dot{q} = 0. \tag{6}$$

Combining equations (5) and (6), the control system $\dot{x} = f_v(x) + g_v(x)u_v$ is obtained. The final expression can be found in [6].

C. Domains and Guards

The domain D_v is the set of admissible configurations $(q, \dot{q}, u_v)$ for vertex v. It is defined by two inequalities:

$$D_v = \left\{ (q, \dot{q}, u_v) \in X \times U_v : \begin{array}{r} R_v^T F_v(q, \dot{q}, u_v) \geq 0 \\ h_v(q, \dot{q}, u_v) \geq 0 \end{array} \right\}. \tag{7}$$

R_v contains the coefficients of the normal reaction forces and the static friction conditions [6]. The purpose of the force-related inequality is to guarantee that some particular parts of the feet are in contact with the ground (i.e. have non-negative normal forces) and are not slipping (i.e. have sufficient friction forces).

Further, unilateral constraints $h_v(q, \dot{q}, u_v)$ can be defined for any part of the model. They are often heights above the ground for different parts of the feet.

The guard S_e is a boundary to D_v. This corresponds to at least one of the properties of the inequalities reaching the value zero and while still decreasing:

$$D_v = \left\{ (q, \dot{q}, u_v) \in X \times U_v : \begin{array}{r} H_e(q, \dot{q}, u_v) = 0 \\ \dot{H}_e(q, \dot{q}, u_v) < 0 \end{array} \right\}, \tag{8}$$

where $H_e(q, \dot{q}, u_v)$ contains the appropriate elements from $R_v^T F_v(q, \dot{q}, u_v)$ and/or $h_v(q, \dot{q}, u_v)$ [8]. For bipedal robots, those elements are either vertical reaction forces (when parts of the feet are lifted from the ground) or heights (when parts of the feet strike the ground).

D. Discrete Dynamics

The discrete transitions of an edge $e \in E$ are defined by the reset map Δ_e. It maps the pre-impact states x^- in D_v to the post-impact states x^+ in D_{v^+}. The configuration is always invariant, meaning the values of the base frame and all joints in $q \in \mathcal{Q}$ cannot change over the transition's infinitesimal time step:

$$q^- = q^+. \tag{9}$$

The velocities, however, are allowed to make jumps. Study the strike impact. The swing foot has a velocity downwards, but when the foot strikes the ground, the vertical velocity immediately becomes zero (given that no slipping or rebound occurs). This creates a discontinuity. Such impacts are assumed to be perfectly plastic, meaning they have to fulfill the plastic impact equation [6]:

$$\begin{bmatrix} M(q^-) & -J_{v^+}^T(q^-) \\ J_{v^+}(q^-) & 0 \end{bmatrix} \begin{bmatrix} \dot{q}^+ \\ \delta F_v \end{bmatrix} = \begin{bmatrix} M(q^-)\dot{q}^- \\ 0 \end{bmatrix}. \tag{10}$$

Worth noting is that the velocities are allowed to take jumps in the discrete transitions, but they do not have to. Lifting a foot from the ground does not cause any discontinuities and the velocity still fulfills the invariance condition: $\dot{q}^- = \dot{q}^+$.

E. Virtual Constraints

The crucial constraints for generating gaits are the virtual constraints, also termed outputs. For each vertex v there are one relative degree 1 scalar $y_{1,v}$ and one relative degree 2 vector $y_{2,v}$. They are defined as the difference between actual and desired outputs [8]:

$$y_{1,v}(q,\dot{q}) = \dot{y}^a_{1,v}(q,\dot{q}) - y^d_{1,v}(\alpha_v), \tag{11}$$

$$y_{2,v}(q) = y^a_{2,v}(q) - y^d_{2,v}(\tau(q),\alpha_v). \tag{12}$$

$y_{1,v}$ is velocity-modulating and its desired path is constant:

$$y^d_{1,v} = v_d. \tag{13}$$

The vector $y_{2,v}$ is position-modulating. Each vector element is an output $o \in O_v$, which desired trajectory is a Bézier polynomial of degree M:

$$y^d_2(\tau,\alpha_o) := \sum_{k=0}^{M} \alpha_o[k] \frac{M!}{k!(M-k)!} \tau^k (1-\tau)^{M-k}. \tag{14}$$

τ is a state-based parameterization of time [8], taking values from 0 to 1 throughout one phase.

F. Partial Hybrid Zero Dynamics

The goal is to drive the virtual constraints towards zero exponentially $(y_{1,v}, y_{2,v}) \to 0$. The feedback linearization control law from equation (28) in [9] is implemented to make the zero dynamics submanifold invariant within all the continuous domains [8].

In the discrete transitions, the configuration q is kept invariant but discontinuities in the velocities $\dot{q}$ are allowed. In terms of the virtual constraints, it corresponds to only requiring the position-modulating outputs $y_{2,v}(q)$ to be invariant through the impacts [7].

This motivates the definition of the partial zero dynamics surface:

$$\mathcal{PZ}_v = \{(q,\dot{q}) \in X \mid y_{2,v}(q) = 0, \dot{y}_{2,v}(q,\dot{q}) = 0\}. \tag{15}$$

$\mathcal{PZ}_v$ is said to be an impact invariant submanifold, if there exists parameters v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$ such that the reset map $\Delta_e(x)$ from $x \in S_e \cap \mathcal{PZ}_v$ ends up in $\mathcal{PZ}_{v^+}$ [8].

Further, the union $\mathcal{PZ} = \cup_{v\in V}\mathcal{PZ}_v$ is hybrid invariant if it is invariant over all continuous phases $v \in V$ and impact invariant over all discrete transitions $e \in E$. In other words, solutions $(q,\dot{q})$ that start in $\mathcal{PZ}$ will stay in $\mathcal{PZ}$ throughout the whole cyclic graph. If so, it is said that the hybrid control system has a partial hybrid zero dynamics (PHZD) [8].

III. Methodology

The goal of this project is to find stable joint trajectories that give rise to a human-like gait.

To summarize the background section: for each phase, denoted by a vertex $v \in V$, the optimal v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$ need to be found. They parametrize the desired output trajectories, which should fulfill all constraints defined in the domain. This includes holonomic, force-related and unilateral as well as additional physical constraints. To generate stable trajectories, the parameters also need to make each $\mathcal{PZ}_v$ impact invariant, i.e. give the hybrid control system a PHZD. The final results, the joint trajectories, are simply the movements of the individual joints required to drive the actual output trajectories towards the desired ones.

A. FROST

AMBER Lab has developed FROST (Fast Robot Simulation and Optimization Toolkit) for MATLAB. It provides tools for all three steps of the implementation: modeling, trajectory optimization and simulation.

The main task of this project is to set up the hybrid control systems correctly. FROST uses that information to generate optimal symbolical expressions for all future calculations. This makes the software powerful. It significantly improves the runtime of both optimization and simulation [4].

FROST provides an optimal trajectory planning function, i.e. a function that finds the optimal parameters v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$. It uses advanced direct collocation algorithms, about which details can be found in [4]. The continuous phases are uniformly discretized and the optimization problem is solved separately at each node. Then continuous curves are fitted to the node solutions, generating values for v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$. They are iteratively improved, until the optimal parameters are found or the maximum number of iterations is exceeded.

The user decides the number of nodes. There is a trade-off: the fewer nodes, the shorter runtime, but too few nodes may not restrict the fitted curve enough and can result in unreasonable and divergent trajectories.

B. Modeling

The amputee-prosthesis model is shown in Fig. 3. from [7], but with the springs removed. Roll joints are denoted ϕ, pitch joints θ and yaw joints ψ. The model has 21 degrees of freedom and the coordinates $q \in \mathcal{Q} \subset \mathbb{R}^{21\times 1}$ consist of:

- 6 base coordinates $R_b = \{x_b, y_b, z_b, \phi_b, \theta_b, \psi_b\}$
- 3 waist joints $q_w = \{\phi_w, \theta_w, \psi_w\}$
- 6 joints in left leg $q_l = \{\phi_{lh}, \theta_{lh}, \psi_{lh}, \theta_{lk}, \phi_{la}, \theta_{la}\}$
- 6 joints in right leg $q_r = \{\phi_{rh}, \theta_{rh}, \psi_{rh}, \theta_{rk}, \phi_{ra}, \theta_{ra}\}$

Since symmetric gaits are made, it does not matter if the right or left leg is the stance leg. After each step cycle, a relabeling occurs, mirroring the model: left $\leftrightarrow$ right. Therefore, the stance leg will always be defined as the right leg.

To set up a hybrid system in FROST, the contacts for each vertex has to be defined. When adding a contact, holonomic constraints are automatically generated. A planar contact

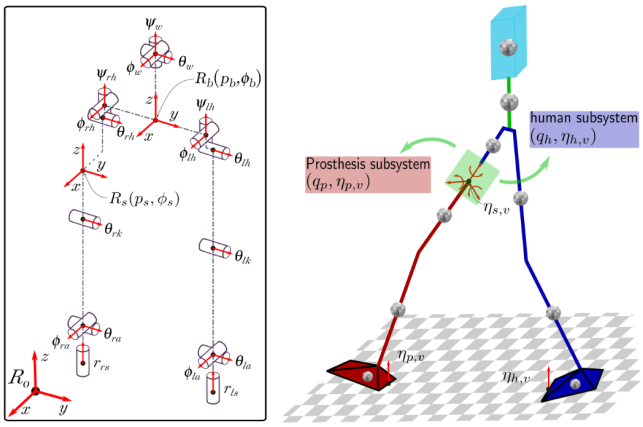

Fig. 3. The amputee-prosthesis model is a chain of links and joints [7]. It has 21 degrees of freedom, since the springs r_{rs} and r_{ls} has been removed.

gives six constraints, position and rotation has to be constant in all directions, while a linear contact only gives five, since it allows rotation around one axis.

We have the three possible contacts of each foot: the foot sole (planar), the toe (linear) and the heel (linear).

To have full ranked matrices and be able to perform all necessary calculations, the following needs to be fulfilled: # virtual constraints + # holonomic constraints = # degrees of freedom. For the human-like gaits, the velocity-modulating actual output is always the linearized hip position:

$$y^a_{1,v} = \delta p_{hip}(q) \tag{16}$$
$$= L_a\theta_{ra} + (L_a + L_c)\theta_{rk} + (L_a + L_c + L_t)\theta_{rh}, \tag{17}$$

where L_a, L_c and L_t denote the length of the ankle, calf and thigh link, respectively. In equation (11), $y^a_{1,v}$ is differentiated. It becomes the linearized hip velocity, which is enforced to be constant = v_d.

The position-modulating actual outputs are linear combinations of the coordinates in $q \backslash R_b \in \mathbb{R}^{15\times 1}$. All vertices have a different amount of holonomic constraints, and therefore a different choice of virtual constraints. They can be chosen in various ways, as long as they are linearly independent. Heuristics led to this batch:

1. Stance ankle pitch $y^a_{sap} = \theta_{ra}$
2. Stance knee pitch $y^a_{skp} = \theta_{rk}$
3. Stance torso pitch $y^a_{stp} = -\theta_{ra} - \theta_{rk} - \theta_{rh}$
4. Stance ankle roll $y^a_{sar} = \phi_{ra}$
5. Stance torso roll $y^a_{str} = -\phi_{ra} - \phi_{rh}$
6. Stance hip yaw $y^a_{shy} = \psi_{rh}$
7. Waist roll $y^a_{wr} = \phi_w$
8. Waist pitch $y^a_{wp} = \theta_w$
9. Waist yaw $y^a_{wy} = \psi_w$
10. Non-stance knee pitch $y^a_{nskp} = \theta_{lk}$
11. Non-stance slope $y^a_{nsl} = -\theta_{ra} - \theta_{rk} - \theta_{rh} + \frac{L_c}{L_c+L_t}\theta_{lk} + \theta_{lh}$

⋮

12. Non-stance leg roll $y^a_{nslr} = \phi_{ra} - \phi_{la}$
13. Non-stance foot roll $y^a_{nsfr} = z_{lfi} - z_{lfo}$
14. Non-stance foot pitch $y^a_{nsfp} = z_{lft} - z_{lfh}$
15. Non-stance foot yaw $y^a_{nsfy} = y_{lft} - y_{lfh}$

To fully control the outputs, # actuators = # virtual constraints is needed. However, the amputee-prosthesis model has a maximum of 15 actuators, since it has 15 joints. This makes one of the multi-contact domains under-actuated.

The triggers for the guards are either the vertical reaction force or the height above the ground for a sole/toe/heel.

The hybrid control systems for each gait are drawn in Fig. 4., Fig. 5. and Fig. 6. Inspiration was taken from [5] and [8]. Details about the specific domains are presented below.

Single support domain

- 1 planar contact: right sole → 6 holonomic constraints.
- 1 velocity-modulating virtual constraint: δp_{hip}.
- 14 position-modulating virtual constraints: no. 2-15.
- 15 actuators: all joints $\{q_w, q_l, q_r\}$.

Double support domain

- 2 planar contacts: right sole and left sole → 12 holonomic constraints.
- 1 velocity-modulating virtual constraint: δp_{hip}.
- 8 position-modulating virtual constraints: no. 2-9.
- 9 actuators: the waist and right leg joints $\{q_w, q_r\}$.

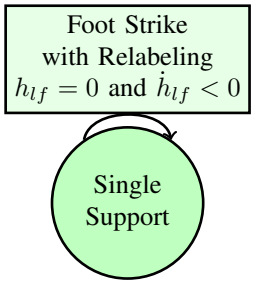

Fig. 4. The hybrid system for single domain gait, with one domain and one transition. h is height above the ground. The indexes stand for left foot.

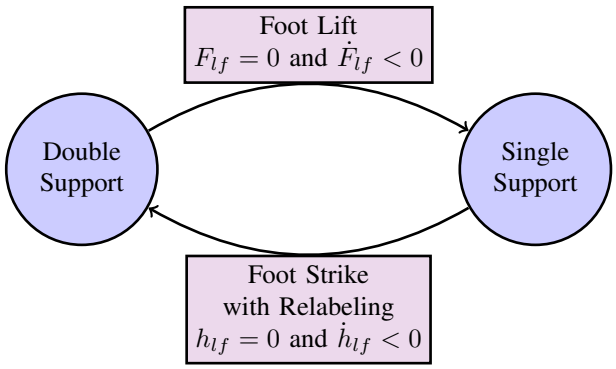

Fig. 5. The hybrid control system for flat-foot gait, with 2 domains and 2 transitions. h is height above the ground and F is vertical contact force. The indexes stand for right/left foot.

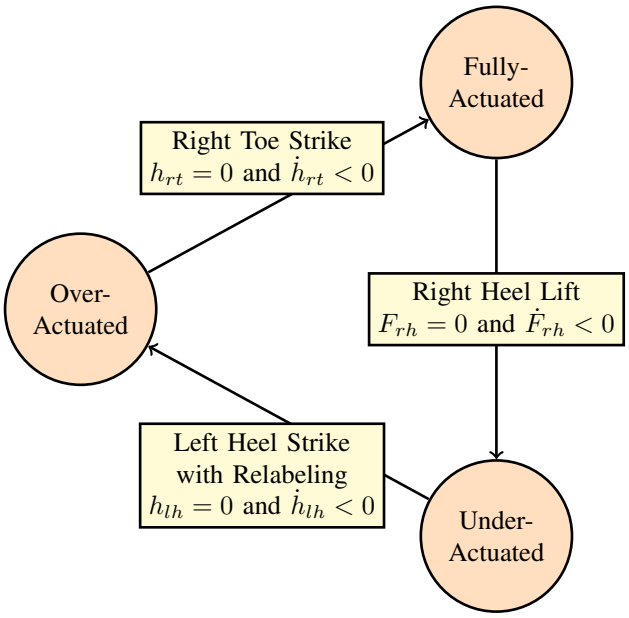

Fig. 6. The hybrid control system for multi-contact gait, with 3 domains and 3 transitions. h is height above the ground and F is vertical contact force. The indexes stand for right/left heel/toe.

Over-actuated domain

- 2 linear contacts: left toe and right heel $\rightarrow$ 10 holonomic constraints.
- 1 velocity-modulating virtual constraint: δp_{hip}.
- 10 position-modulating virtual constraints: no. 1-10.
- 11 actuators: the waist and right leg joints plus the left knee pitch and left ankle pitch $\{q_w, \theta_{lk}, \theta_{la}, q_r\}$.

Fully-actuated domain

- 1 planar contact: right sole $\rightarrow$ 6 holonomic constraints.
- 1 velocity-modulating virtual constraint: δp_{hip}.
- 14 position-modulating virtual constraints: no. 2-15.
- 15 actuators: all joints $\{q_w, q_l, q_r\}$.

Under-actuated domain

- 1 linear contact: right toe $\rightarrow$ 5 holonomic constraints.
- 1 velocity-modulating virtual constraint: δp_{hip}.
- 15 position-modulating virtual constraints: no. 1-15.
- 15 actuators (= under-actuated): all joints $\{q_w, q_l, q_r\}$.

C. Trajectory optimization

A heuristic assumption is that humans walk in the most efficient way possible. To implement, a cost function is added: the total torque of the joints. The FROST optimization method then tries to find trajectories that minimize the cost function (i.e. minimize energy) while still fulfilling all constraints.

Before running the optimization, limits for a number of values are set. They help the program find a convergent gait, but they can also impede the optimization if they are chosen poorly. The most influential are the linearized hip position and the step length, which have to be chosen wisely.

The output is the parameters v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$.

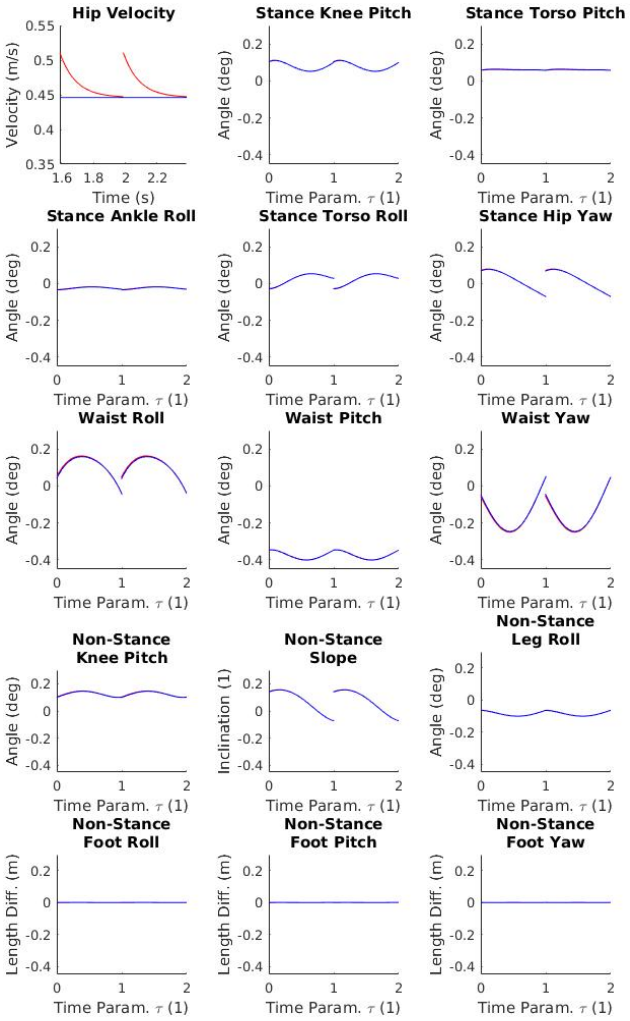

Fig. 7. Desired (blue) and actual (red) output trajectories for single domain gait. Step number 5 and 6 of a long step sequence are plotted.

D. Simulation

Lastly, the simulation function in FROST is called. The output is trajectories for a number of variables, including the joint trajectories that are sought.

IV. Results

For single domain gait, Fig. 7. confirms that desired output trajectories (blue lines) have been generated and that the simulation is able to follow them (red lines). The periodic phase portraits in Fig. 8. validates that the optimization is solving the right problem, one that results in a convergent gait. When running the animation, the gait looks human-like, see snapshots in Fig. 9.

Stable joint trajectories have not yet been obtained for the other, more complex gaits. The optimization method in FROST does not converge, i.e. it does not find a solution that satisfies all the necessary constraints.

If the periodicity condition is disregarded, the optimization method can find a solution over two domains. When watching the animation, the resulting parameters v_d and $\{\{\alpha_o\}_{o\in O_v}\}_{v\in V}$ give a motion that looks human-like, see

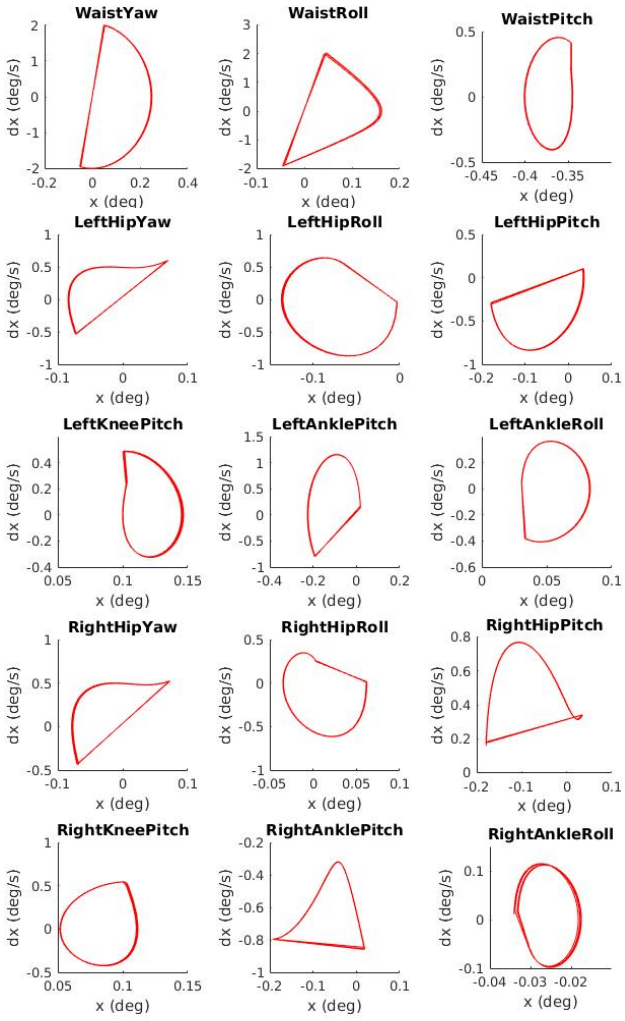

Fig. 8. Stable phase portraits for single domain gait. The 10 first steps of a long step sequence are plotted.

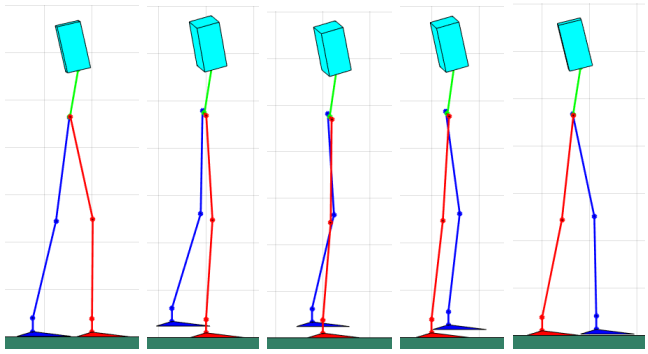

Fig. 9. Snapshots from the animation of one step in single domain gait.

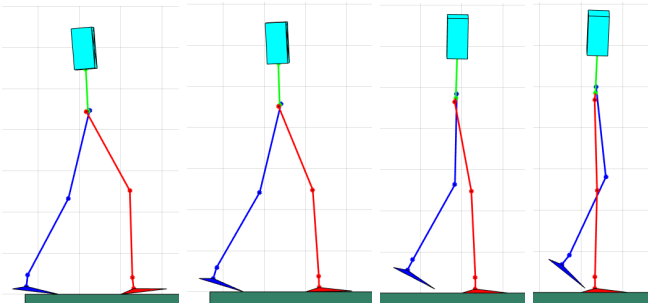

Fig. 10. Snapshots from the animation of Over-actuated domain to Fully-actuated domain in multi-contact gait.

snapshots in Fig. 10. The simulation function does however generate divergent joint trajectories and the animated robot then falls over.

A. Discussion

Using FROST, each implementation is built up by multiple MATLAB scripts. There is one script for the hybrid control system, one for each domain, one for each guard etc. It is therefore easy to modify the implementation. One building block can be replaced or updated without messing up the rest of the code too much. This is an important advantage of FROST.

This particular project was limited to 10 weeks. Within this time, all hybrid systems were set up, but some optimization methods could not find a solution. This work continues in the AMBER lab.

There are several possible explanations to why the optimization methods do not converge. First of all, the limits may not be chosen properly. They are necessary because they specify where to search for solutions, but as mentioned in section III. C., they might also block the optimization from finding a solution.

For some limit choices, the optimization reaches very small errors. Such small errors usually lead to convergence, but, for some reason, not in this case. So far, Bézier polynomials of degree 4 have been used in all implementations. A recent theory is that this is too restrictive for the output trajectories of flat-foot and multi-contact gait. Therefore, degree 6 polynomials are currently implemented instead.

Alternatively, degree 4 might not be restrictive enough. Study the double support domain. It is a short domain where basically all that happens is that the hip moves forward. Both feet are glued to the ground. A Bézier of degree 4 seems to give an unnecessarily amount of freedom for such basic joint movements. It might be advantageous to vary the degree of the polynomials between the different domains, depending on their complexity.

V. Conclusions

From this project it is concluded that it is appropriate to use the software FROST to model hybrid control systems and generate human-like gaits for AMPRO3. A stable single domain gait has already been found. Regarding flat-foot gait and multi-contact gait, some work still remains.

The final goal is to implement the simulated trajectories on the actual prosthesis and perform experiments.

The model used in this project is symmetric. It would be interesting to generate gaits for an asymmetric amputee-prosthesis model as well, since that is more realistic.

Further, the FROST code would be easy to modify to simulate other walking behaviors for the amputee-prosthesis

model, such as walking on slopes and stair walking.

VI. Acknowledgments

I would like to thank Aaron D. Ames for giving me the opportunity to work and learn in the AMBER Lab. I would also like to thank Rachel Gehlhar and Jacob Reher for their help throughout the project.

References

[1] T. R. Dillingham *et al.*, "Limb amputation and limb deficiency: Epidemiology and recent trends in the United States," *Southern Medical Journal*, 2002.

[2] D. A. Winter, "The Biomechanics and Motor Control of Human Gait: Normal, Elderly, and Pathological," University of Waterloo Press, 1991.

[3] H. Zhao *et al.*, First Steps Toward Translating Robotic Walking To Prostheses: A Nonlinear Optimization Based Control Approach, *Autonomous Robots: Special Issue on Assistive and Rehabilitation Robotics*, 2016.

[4] A. Hereid and A. D. Ames, "FROST*: Fast Robot Optimization and Simulation Toolkit," unpublished.

[5] H. Zhao *et al.*, Human-inspired Multi-Contact Locomotion with AMBER2, in *International Conference on Cyber-Physical Systems*, Berlin, 2014.

[6] H. Zhao, "From Bipedal Locomotion to Prosthetic Walking: a Hybrid System and Nonlinear Control Approach," Ph.D. dissertation, School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA, 2016, ch. 3.

[7] H. Zhao *et al.*, "3D Multi-Contact Gait Design for Prostheses: Hybrid System Models, Virtual Constraints and Two-Step Direct Collocation, in *IEEE 55th Conference on Decision and Control (CDC2016)*, Las Vegas, 2016, pp. 3668-3674.

[8] A. Hereid *et al.*, "3D Dynamic Walking with Underactuated Humanoid Robots: A Direct Collocation Framework for Optimizing Hybrid Zero Dynamics, in *IEEE International Conference on Robotics and Automation (ICRA2016)*, Stockholm, 2016, pp. 1447-1454.

[9] A. D. Ames, "Human-Inspired Control of Bipedal Walking Robots," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1115-1130, May 2014.