Simulation and review of “First Steps Toward Underactuated Human-Inspired Bipedal Robotic Walking”

Gabriel Enrique García Chávez

This file documents the most important parts and conclusion from the simulation of the paper “First Steps Toward Underactuated Human-Inspired Bipedal Robotic Walking” form Amber LAB, reference [2].

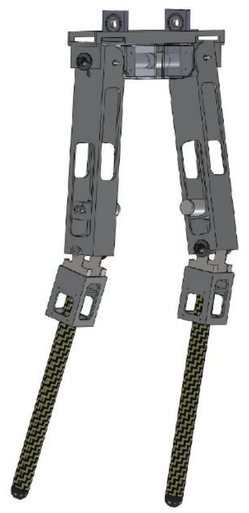


Figure 1: AMBER robot

The important parts of the paper are:

- First, they took data collected from 9 subjects, this is explained in a 2011 paper (Ref [2]), 5 angles from normal walking of people, 1 in ankle, 2 knees, and 2 hips.
- By studying the data, they obtain 5 variables, linear transformations from 5 angles that has low standard deviation from the mean. The new variables represents Hip position, Non-stance leg slope, Stance knee angle, Non-stance knee angle, and Angle of the torso from vertical. The vector of the last 4 variables will be $y_a(\theta)$
- They model Hip position as $\delta p_{hip}^{d}(t,v) = v_{hip}t$, and the other variables as $y_H(t,\alpha) := e^{-\alpha_4 t}(\alpha_1 \cos(\alpha_2 t) + \alpha_3 \sin(\alpha_2 t)) + \alpha_5$. They define the vector $y_d(t)$ as the column vector with the $y_H(t,\alpha)$ respective (each variable has it’s respective α)
- They want to track $y_d(t)$, but they do a special change of variables that will remove the t variable on it. They uses the fact that $\delta p_{hip}^{d}(t,v) = v_{hip}t$, so they define a time “dependent” of the variables:
$$t \approx \frac{\delta p_{hip}}{v_{hip}} \rightarrow \tau(\theta) = \frac{\delta p_{hip}^{R}(\theta) - \delta p_{hip}^{R}(\theta^{+})}{v_{hip}}$$
So they finally define a “*human-inspired output*” as follows:
$$y(\theta) = y_a(\theta) - y_d(\tau(\theta))$$
This is the big difference with common tracking: They are not tracking $y_a(\theta) - y_d(t)$. Ideally, in the best scenario, if the hip holds $\delta p_{hip}^{d}(t,v) = v_{hip}t + \delta p_{hip}^{R}(\theta^{+})$, then $\tau(\theta) = t$ and actually $y_a(\theta) \rightarrow y_d(\tau(\theta)) = y_d(t)$. They will force this in the regression for α as we will see later.

- They track $y(\theta)$ using PFL. The control law follows:

$$u(\theta,\dot{\theta}) = -\left(L_g L_f y(\theta,\dot{\theta})\right)^{-1}\left(L_f^2 y(\theta,\dot{\theta}) + 2\varepsilon L_f y(\theta,\dot{\theta}) + \varepsilon^2 y(\theta)\right)$$

  Let's define $\boldsymbol{Z_\alpha}$ as the Zero Dynamics, it will be:

$$\boldsymbol{Z_\alpha} = \{(\theta,\dot{\theta}) \in TQ_R : y(\theta) = 0, L_f y(\theta,\dot{\theta})\}$$

  Note that $\boldsymbol{Z_\alpha}$ depend on α, because $y(\theta)$ contains a term $y_d(\tau(\theta))$, and the function y_d depends on α:

  But let's realize that $\tau(\theta)$ correspond to the Zero Dynamics, so they have lost control of it. They define ξ_1 as an affine transformation of $\tau(\theta)$. Again, ideally they want $\tau(\theta) = t$, implying that we want ξ grow linearly.

- They also define a function $\vartheta(\alpha) = \theta$ s.t:

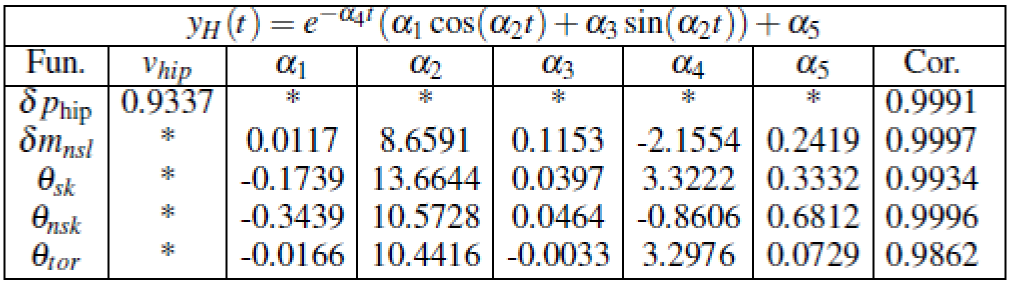
$$\begin{bmatrix} y(\Delta_\theta \theta) \\ h_R(\theta) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}$$

  So basically, $\vartheta(\alpha)$ will give us a point in the space configuration that is in the guard, and most important, that under collision, you will be sent to $y(\Delta_\theta\, \theta) = 0$, a requisite, but not enough, for Zero Dynamics. Although they say that finding ϑ is essentially an Inverse Kinematics problem, it would potentially be a tedious algebraic problem (but could be easily solved by numeric ways, not necessarily closed form, they simply doesn't specify they it's solved). They also define $Y(\theta) = \begin{bmatrix} d\delta p_{hip}^R(\theta) \\ dy(\theta) \end{bmatrix}$ and $\dot{\vartheta}(\alpha) = Y^{-1}(\vartheta(\alpha)) \begin{bmatrix} v_{hip} \\ \mathbf{0} \end{bmatrix}$.
- Here comes the critical part: They don't simply apply regression to the data and find α. They apply regression but with a trick: They restrict the variables to hold some special condition that will lead stability under collision. Let's remember, the robot has 5 links, but 4 actuations, so after Partial Feedback Linearization of any desired variable, there will always be a zero dynamics of 2 variables (counting position and velocities).

$$\alpha^* = \underset{\alpha \in \mathbb{R}^{21}}{\operatorname{argmin}}\ \mathrm{Cost_{HD}}(\alpha) \tag{24}$$
$$\text{s.t}\quad y(\vartheta(\alpha)) = \mathbf{0} \tag{C1}$$
$$dy(\Delta_\theta \vartheta(\alpha))\Delta_{\dot{\theta}}(\vartheta(\alpha))\dot{\vartheta}(\alpha) = \mathbf{0} \tag{C2}$$
$$dh_R(\vartheta(\alpha))\dot{\vartheta}(\alpha) < 0 \tag{C3}$$
$$\mathscr{D}_{\mathbf{Z}}(\vartheta(\alpha)) < 0 \tag{C4}$$
$$0 < \Delta_{\mathbf{Z}}(\vartheta(\alpha)) < 1 \tag{C5}$$

  Specifically, the first 3 restrictions will make the system stay in the Zero Dynamics even after collision. This is equivalent to: $\Delta_R(S_R \cap \boldsymbol{Z_\alpha}) \in \boldsymbol{Z_\alpha}$, where Δ_R is the "Reset Map", called also "Transition Dynamics" and $\boldsymbol{Z_\alpha}$ is the Zero Dynamics, and S_R is the guard for collision. Proof is the equivalence is found at [1] as Theorem 1 in PHZD Optimization. In summary $\Delta_R(S_R \cap \boldsymbol{Z_\alpha}) \in \boldsymbol{Z_\alpha}$, means that, even after one collision, the system will remain in the Zero Dynamics. This is great: The control law allows the system remain in the Zero Dynamics (by definition). And this restriction in the optimization makes that the $\boldsymbol{Z_\alpha}$ will map to itself after touching the ground, so the error will be still 0 in the tracking variables. This not solves the problem with the Zero Dynamics $\tau(\theta)$, but the last 2 restrictions do. In fact, the fourth condition implies the existence of a limit cycle inside of the Hybrid Dynamics Surface and the fifth implies its stability. In the demonstration they use a $\xi = \xi^-$ as Poincare map. So we are ensured that $\tau(\theta)$ (affine to ξ) will grow and will be reset to a lower value, in order to grow again until reaching the equivalent of ξ^- establishing its limit cycle. We have now that $\tau(\theta)$ grows as time grows, but no necessarily $\tau(\theta) = t$. However, this conditions are enough to produce a stable walking in the robot.

So here we are, parameters obtained comes from the optimization, showed in the paper, and extracted here:



$y_H(t) = e^{-\alpha_4 t}(\alpha_1 \cos(\alpha_2 t) + \alpha_3 \sin(\alpha_2 t)) + \alpha_5$							
Fun.	v_{hip}	α_1	α_2	α_3	α_4	α_5	Cor.
δp_{hip}	0.9337	*	*	*	*	*	0.9991
δm_{nsl}	*	0.0117	8.6591	0.1153	-2.1554	0.2419	0.9997
θ_{sk}	*	-0.1739	13.6644	0.0397	3.3222	0.3332	0.9934
θ_{nsk}	*	-0.3439	10.5728	0.0464	-0.8606	0.6812	0.9996
θ_{tor}	*	-0.0166	10.4416	-0.0033	3.2976	0.0729	0.9862

We proceed to simulate this values to watch the reset map, and the behavior of the system. We only have a little problem: The paper doesn't provide the lengths of legs or shanks that were defined for the solver. We can estimate them using the Zatsiorsky-Leyva parameters, we require the height and weight of the person. This parameters allows us to get the weights, lengths, inertias and center of mass of some parts of the body, in particular legs and shank. More information and references about this can be found at [3].

So in ModelSPSym.m, the Dynamics of the system is defined. Note that it uses a program for the Zatsiorsky – Leyva parameters to get the values needed for the system. We have defined the weight in 69 kg, based on "weights ranging from 47.7 kg to 90.9 kg" from the AMBER LAB paper, ref. [2]. Similarly, we have defined the height in 1.74 cm according to "heights ranging from 160.0 cm to 188.5 cm". Note that this can be different from the values they have used, but enough to make the system work.

"Control_Refs_t_Hybrid.m" runs the system using the control law that sends the dynamics to the Zero Dynamics. Here we have some observations:

- We run the solver ode45, taking as the system the robot model "AMBER", u can change the initialization of the system almost randomly (See note 3).
- We stop the solver when a collision in the swing foot to the ground happens.
- We reset the solver, and run it again, this time the initial condition will be the reset maps applied to the last state of the last simulation. We repeat this as many time as it's needed in order to converge to the Hybrid Zero Dynamics Surface. A soft indicator of this is the time that takes the system in order to give a step, but it is enough for this.

- We show only for visualization the robot giving steps, and after that, we present some graphics.

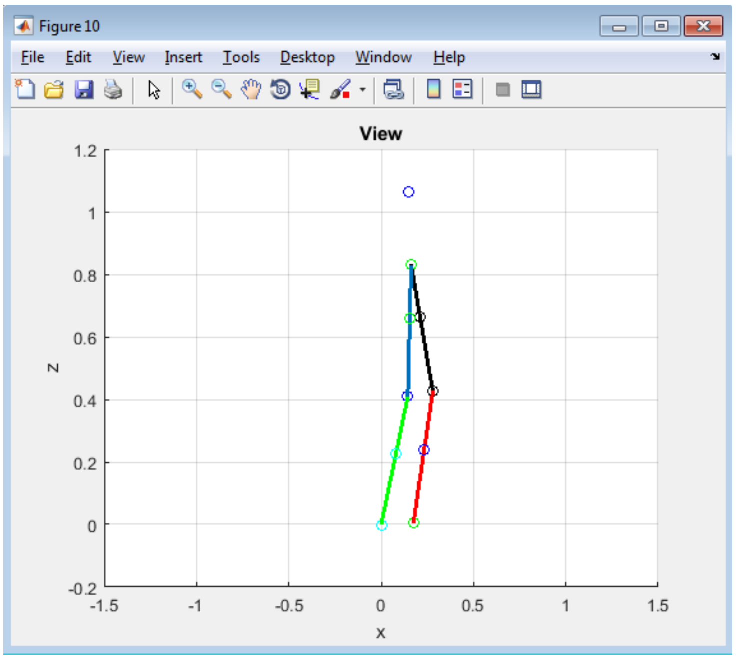


Figure 2: AMBER robot simulation. Preimpact state, the green leg is in stance state.

Figure 6 shows us the value of $y(\theta)$. Let's see that, given the feedback law control, this value will always go to 0. Let's remember that the optimization was done forcing that the reset map applied in the Zero Dynamics will give new values still relying on the Zero Dynamics, this implies$y(\Delta_\theta \theta)$. We are watching the plot of the final limit cycle, so time 0 is post impact, and it must hold: $y\big(\theta(t=0)\big)=0$, but we are not watching that. Reason is simple: We are not matching the values they've used in the optimization. But it's important to note that still find a limit cycle close to the Zero Dynamics.

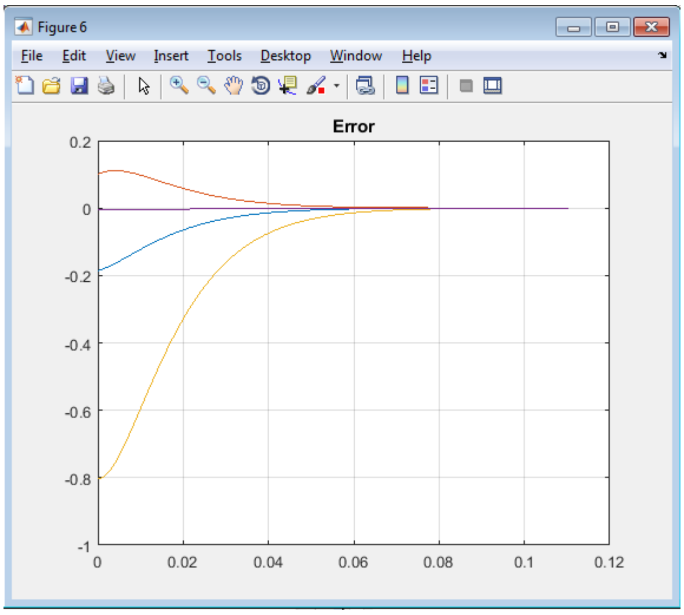


Figure 3: Error dynamics of $y(\theta)$ after reaching a hybrid limit cycle.

Figure 7 shows the values of the differences of $y_a(\theta)$ and the desired values at the time $y_d(t)$. Remember: We are not tracking $y_d(t)$, we are tracking $y_d\big(\tau(\theta)\big)$, and this figure simply shows that $\tau(\theta) \neq t$, otherwise, this errors will be perfectly 0.

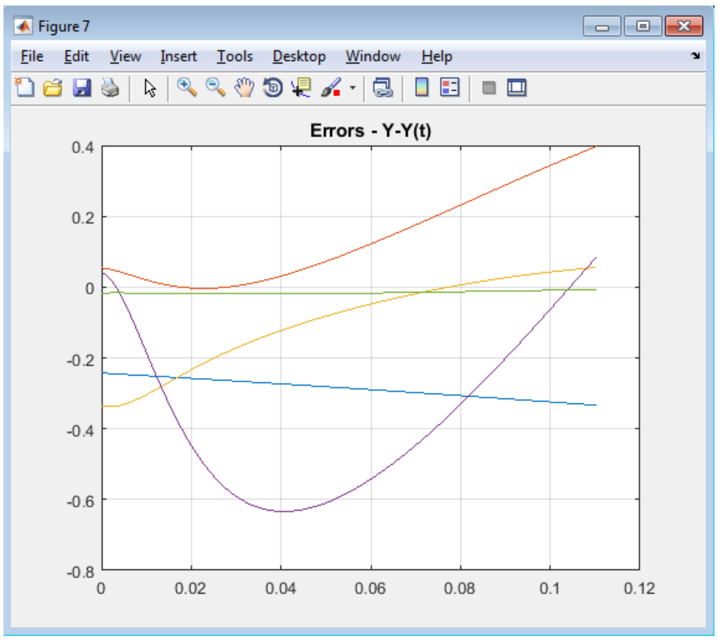


Figure 3: Figure 7 from simulation: Error dynamics of $y_a(\theta) - y_d(t)$.

Figure 9 gives us more information of $\tau(\theta)$. Remember that this is closely related to the hip position, so we plot this value vs time.

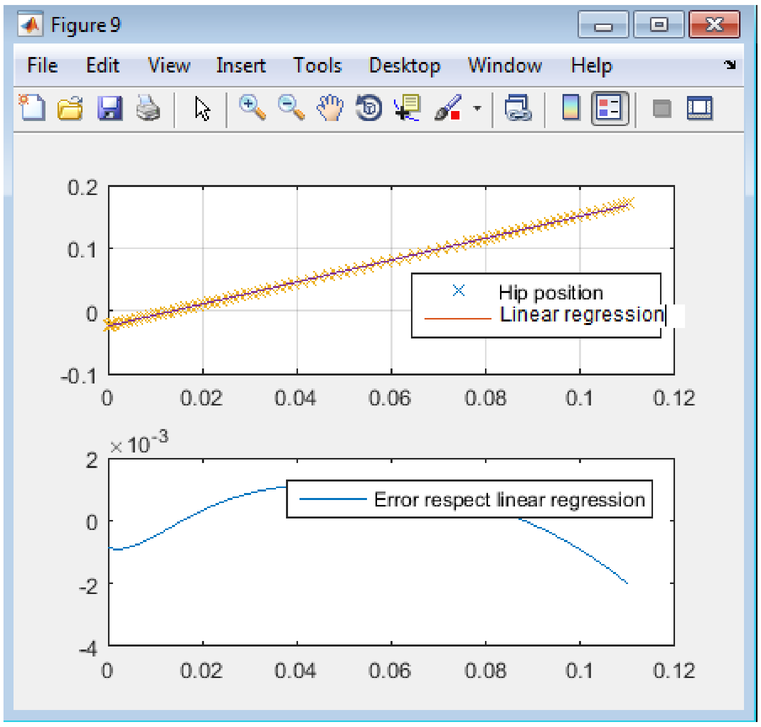


Figure 4: Hip position and it's linear regression respect time

In fact:

$$\tau(\theta) = \frac{\delta p_{hip}(\theta) - \delta p_{hip}(\theta^+)}{v_{hip}} \approx \frac{1.7500t}{0.9337} \approx 1.8743\ t$$

Finally we have that $\tau(\theta) \approx 1.8743\ t$. Although finally we have $\tau(\theta) \neq t$. We have an important fact: $\tau(\theta)$ is proportional to time. The control is designed in order to make $y_a(\theta)$ track $y_d\big(\tau(\theta)\big)$, so we are tracking $y_d(1.8743\ t)$, instead of $y_d(t)$, which means that we are tracking the desired trajectory, with the exception that we are making it almost two times faster.

Again, we are not using the physical parameters they used for the system, so we are not waiting it to match perfectly. But even if we use their parameters, there is no security that we will have $\tau(\theta) \approx t$, let's remember that their demonstration shows that there exist an hybrid limit cycle inside of the Zero Dynamics Surface, which only means that $\tau(\theta)$ grows as time grows. We could wait that result indirectly: If the values to fit implies that in a biped, no ankle force is applied, which means humans apply a minimum amount of torque at stance ankle.

Finally let's see that, we are obtaining a stable limit cycle even with other parameters. This is really amazing given the fact that, if we want to track directly the angles $y_d(t)$ we will only obtain a falling robot. You can see this by changing in the solver the system from AMBER to AMBERt. So, tracking this "human-inspired output" is useful to track even robots with little parameter changes, those for which the optimization was not exactly designed.

References:

[1] A. D. Ames. "First steps toward automatically generating bipedal robotic walking from human data". In 8th International Workshop on Robotic Motion and Control, Poland, 2011.

[2] A. D. Ames. "First Steps Toward Underactuated Human-Inspired Bipedal Robotic Walking". In: Robotics and Automation (ICRA), 2012 IEEE International Conference on. IEEE, 2012.

[3] P. A. Miranda-Pereira and L. P. Milian-Ccopa, "Brief biomechanical analysis on the walking for a lower-limb rehabilitation exoskeleton". In: Robotics and Intelligent Sensors (IRIS), 2016 IEEE International Symposium on. IEEE, 2016.

Additional notes on the simulation:

*Note 1: This is in general, a kinematics and dynamics model Generator. If you have other parameters for the AMBER robot, or any other serial manipulator you can obtain the model and perform the tracking of the human inspired output by running the following sequence:

Run “ModelSPSym.m”

Run “faith.m”

Run “GenController.m”

Run “GenGuard.m?

*Note 2: If you are having problems with optset in a new model, run “Control_Refs_t_Hybrid_old.m”. You will need to adapt “eqPspm.m” from your new “eqPsp.m”

*Note 3: If you initialize the system in other initial condition, give the solver enough time to allow the robot give a step. After many steps, limit cycle will be reached by the biped.