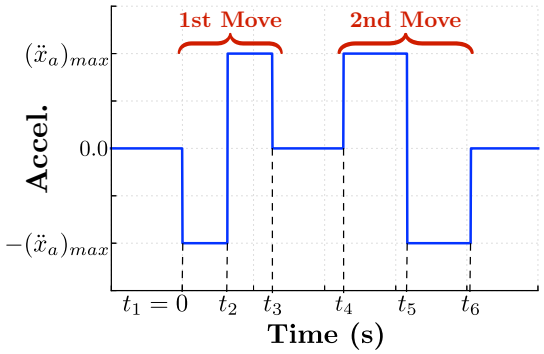
III Using Input Shaping to Validate RL Controllers

In utilizing RL to define a control strategy for a robotic system, the resulting commands sent to the system are often described as optimal, or at least approaching optimal. They are described as such due to the nature of RL problems in that the techniques used to learn a policy are optimization theory based, so the policy being trained is one that is approaching an optimal solution in regards to the reward defined. Interpreting the commands claimed to be optimal, in the context of control theory, is an important part of utilizing RL for defining control strategies for robotic systems. In the case of a flexible jumping robot, where the command is to jump the the system as high as possible or as efficiently as possible, the questions arises if the the commands generated truly approaching an optimal solution or not. To answer this question, the commands resulting from the trained agents can be analyzed leveraging conventional control theory. Methods such as input shaping have been shown to be effective for defining optimal control strategies for flexible jumping systems and can be used to evaluate the commands generated by the RL generated policies [2].

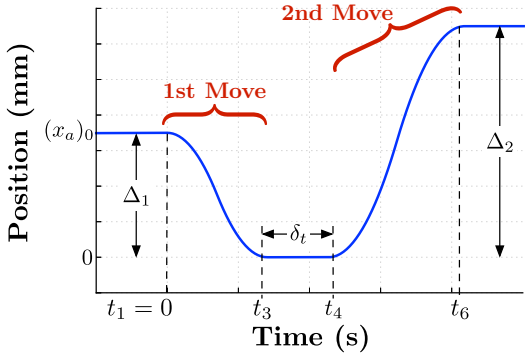
3.1 Input Shaping Controller Input

Bang-bang-based jumping commands like the one shown in Figure 14a are likely to result in a maximized jump height [2]. For these command types, regarding the monopode jumping system, the actuator mass travels at maximum acceleration within its allowable range, pauses, then accelerates in the opposite direction. Commands designed to complete this motion are bang-bang in each direction, with a selectable delay between them. The resulting motion of the actuator along the rod is shown in Figure 14b. Starting from an initial position, x_{a_0}, the actuator moves through a motion of stroke length Δ_1, pauses there for δ_t, then moves a distance Δ_2 during the second portion of the acceleration input.

This bang-bang-based profile can be represented as a step command convolved



(a) Jumping Command [2]



(b) Resulting Actuator Motion [2]

Figure 14. Jumping Command Profiles

less space than other figures?

with a series of impulses, as is shown in Figure 15 [41]. Using this decomposition, input-shaping principles and tools can be used to both analyze and design the impulse sequence [42, 43]. For the bang-bang-based jumping command, the amplitudes of the resulting impulse sequence are fixed, $A_i = [-1, 2, -1, 1, -2, 1]$. The impulse times, t_i, can be varied and optimized leading to a maximized jump height of the monopode system [2]. Commands of this form will often result in a stutter jump, like what was shown in Figure 7b of Chapter 2, where the small initial jump allows the system to compress the spring to store energy to be used in the final jump.

shown in Figure 15

3.2 Review of Vector Diagrams

One tool that seeks to simplify the design of impulse sequences for input shapers is the vector diagram [43]. The vector diagram represents the vibration caused by an impulse as a vector, and the vibration induced by an impulse sequence as the sum of the sequence's representative vectors. As such, the vector diagram can provide a visual means of both analysis and design of impulse sequences.

The process of plotting an impulse sequence on a vector diagram is shown in Figure 16. Each impulse is plotted on the vector diagram in polar coordinates, where the magnitude of each vector is simply the impulse magnitude, and its angle is $\theta = \omega t_i$, where ω is the system natural frequency, and t_i is the time location of the impulse.

To calculate the residual vibration caused by a sequence of impulses, the

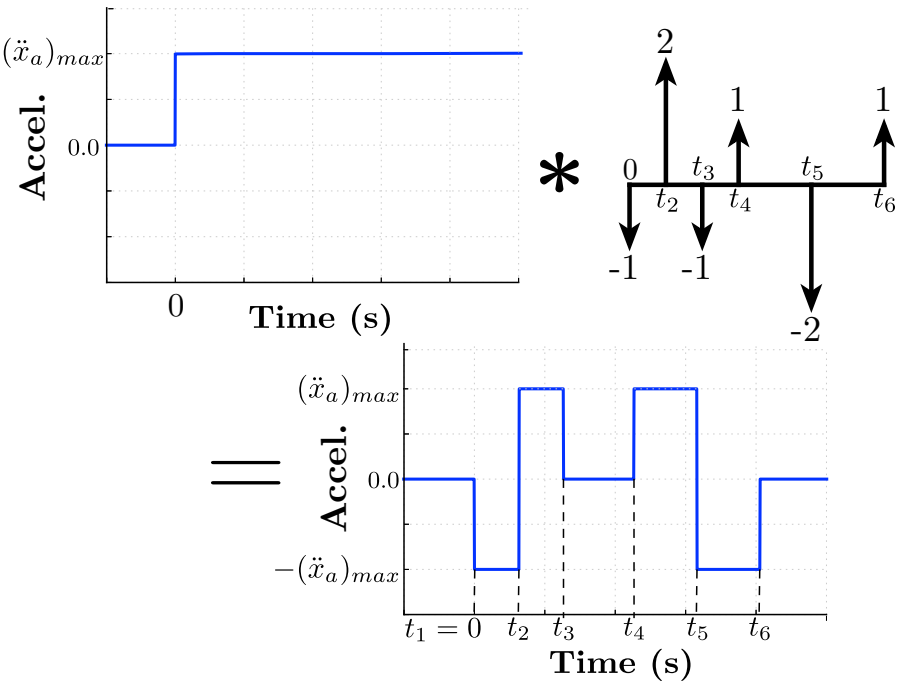


Figure 15. Decomposition of the Jump Command into a Step Convolved with an Impulse Sequence [2]

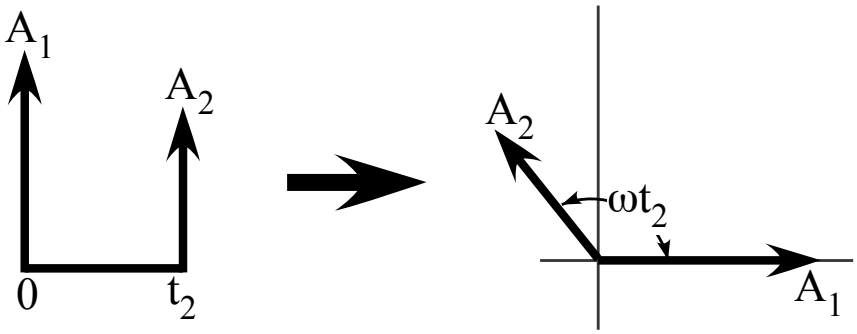


Figure 16. Plotting Impulses on a Vector Diagram

representative vectors are summed, as shown in Figure 17. The magnitude of the residual vibration caused by the sequence is proportional to the magnitude of the resultant vector, A_R.

The vector diagram can also be used as an input shaper design tool. For example, a third vector can be added to the sequence plotted in Figure 17 to produce zero vibration. This third vector, A_3, could be placed opposite of A_R, as shown in Figure 18a. When A_3 is placed this way, the three vectors in the diagram sum to zero,

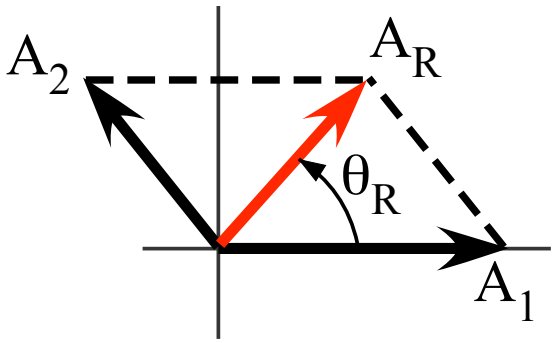
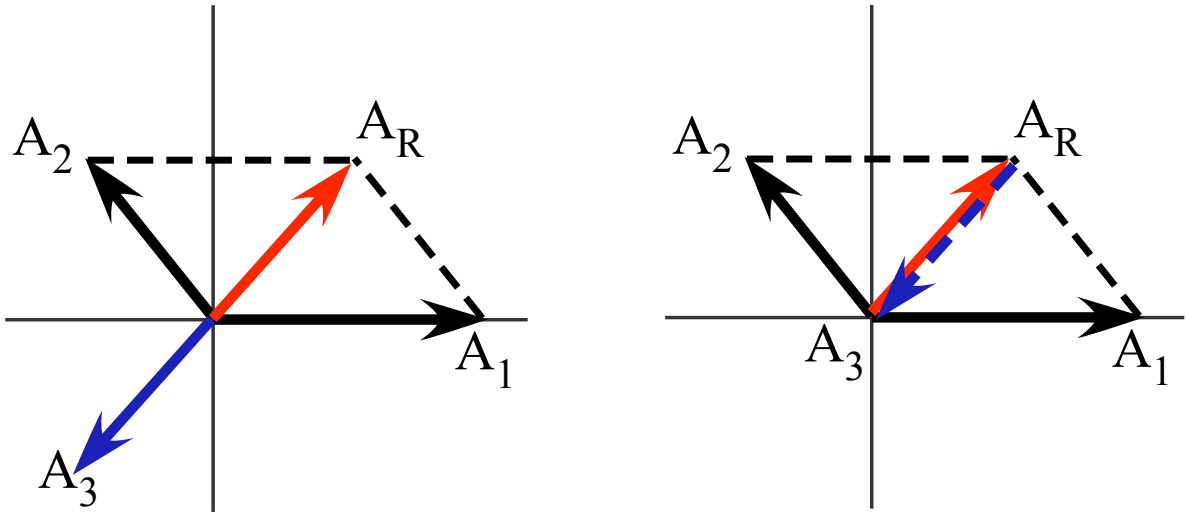


Figure 17. Resultant Vibration Vector from Adding Impulses



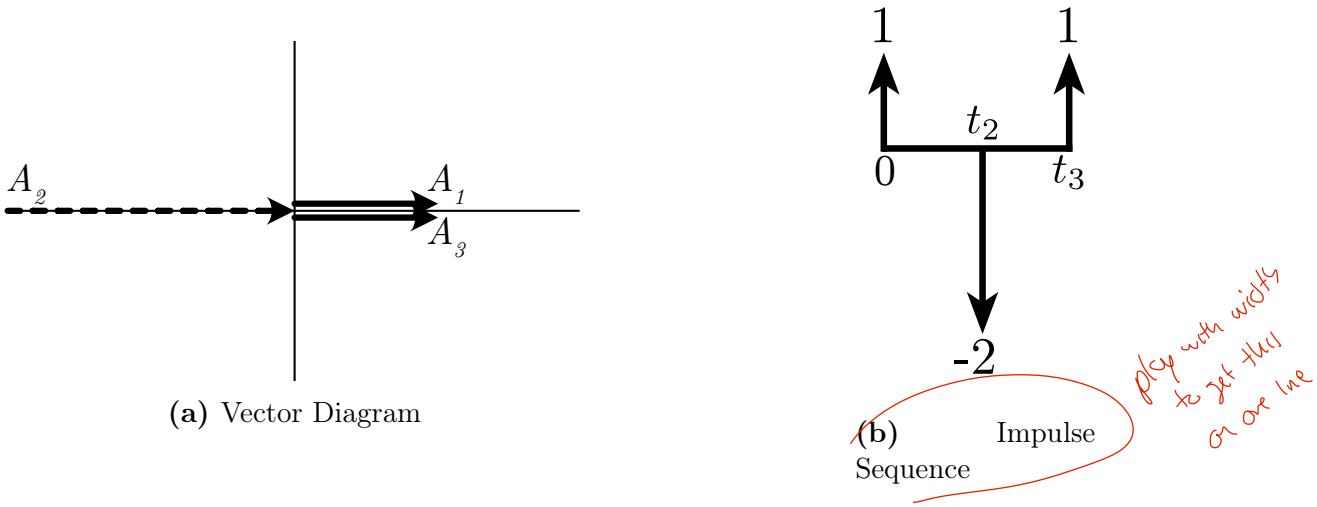
(a) Adding a Positive Impulse **(b)** Adding a Negative Impulse

Figure 18. Designing Input Shapers Using Vector Diagrams

indicating the impulse sequence will excite zero vibration at the design frequency and damping ratio.

To plot a negative impulse, the vector simply points toward the origin instead of away. Another option to design a zero vibration shaper from Figure 17 is to place a negative impulse at A_R, as is shown in Figure 18b.

In the context of this work, the vector diagram can be used to design an impulse sequence that *maximizes* vibration. In other words, instead of arranging vectors to force A_R toward zero, the vectors can be arranged to increase its magnitude.



(a) Vector Diagram

(b) Impulse Sequence

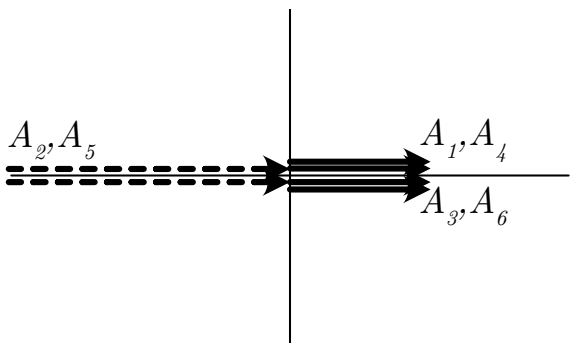
Figure 19. Maximum Vibration Bang-bang Impulse Sequence

3.3 Analysis of Learned Jumping Commands

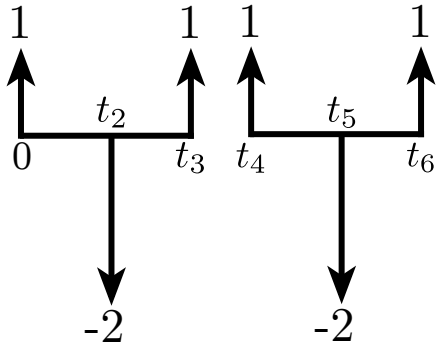
In this section, the learned stutter-jumping commands for maximum jump height from Chapter 2 will be analyzed using tools traditionally used for input shaping. This analysis will focus on the command shown previously in Figure 12b. At first glance, the Height Agent command from this plot shares some similarities with the command introduced in Figure 14a. It is approximately two bang-bang commands.

A vector diagram of a single bang-bang command that maximizes vibration is shown in Figure 19a. The three impulses add constructively and indicate that a bang-bang command has the potential to excite three times the vibration of a single, unity-magnitude impulse (which is typically used as a proxy for the original reference command). The impulse sequence that is represented by the vector diagram is shown in Figure 19b. The spacing of the impulses is such that the second, negative impulse should occur at $t_2 = \tau/2$ and the final impulse, t_3, at time τ, where τ represents the system period ($\tau = 2\pi/\omega$).

A vector diagram for a series of two bang-bang commands designed to maximize vibration is shown in Figure 20a. This second bang-bang command in the sequence is just a repeat of the first, but beginning at time τ, rather than at zero. The resulting



(a) Vector Diagram



(b) Impulse Sequence

Figure 20. Maximum Vibration Dual Bang-bang Impulse Sequence

impulse sequence is shown in Figure 20b and can be written as:

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} 1 & -2 & 1 & 1 & -2 & 1 \\ 0 & \frac{\tau}{2} & \tau & \tau & \frac{3\tau}{2} & 2\tau \end{bmatrix} \tag{16}$$

where A_i and t_i represent the i^{th} impulse amplitude and time, respectively. Given that the third and fourth impulses occur at the same time, the impulse sequence can further be simplified to:

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} 1 & -2 & 2 & -2 & 1 \\ 0 & \frac{\tau}{2} & \tau & \frac{3\tau}{2} & 2\tau \end{bmatrix}. \tag{17}$$

The convolution of this impulse sequence with a step command in actuator acceleration, matching the process from Figure 15, is shown in Figure 21. ~~This command represents one that induces the optimal jump height for a linearized version of the monopode system.~~ The resulting jumping response is shown in Figure 22. This case represents the upper bound of the jump height possible for a linearized version of the monopode system, given the actuator limits imposed. While the form of the command in Figure 21 is similar to that learned by the RL algorithm, shown in Figure 12b, the timing of the transitions between negative and positive acceleration are slightly different. This results in the RL algorithm not quite achieving the theoretical upper bound.

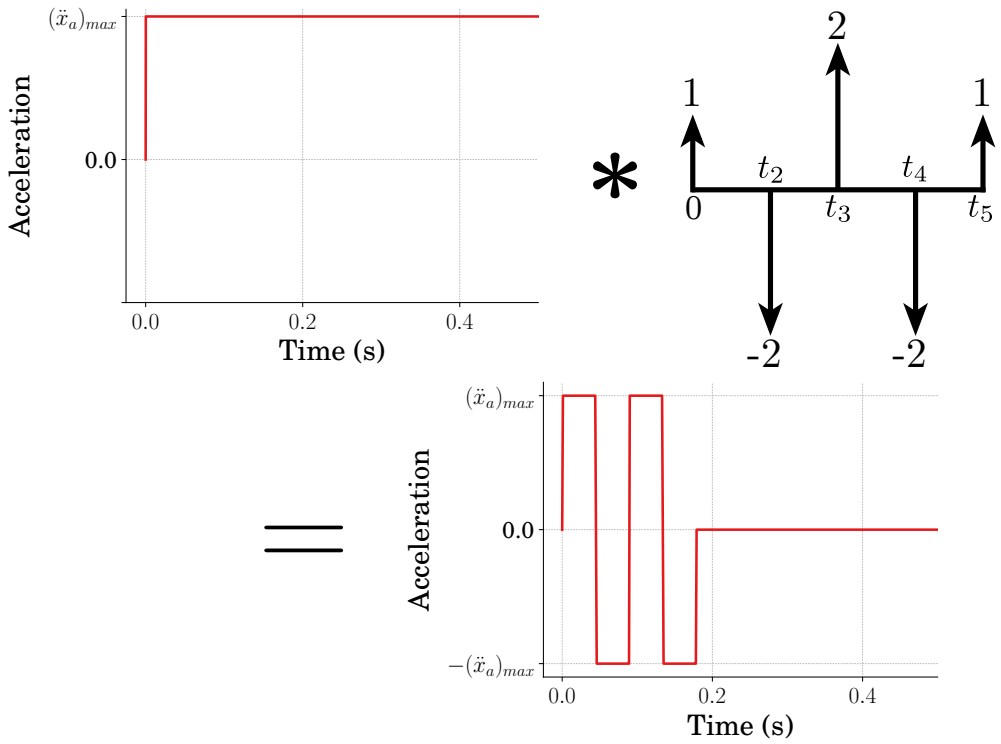


Figure 21. Convolution of Maximum Vibration Bang-bang Impulse Sequence

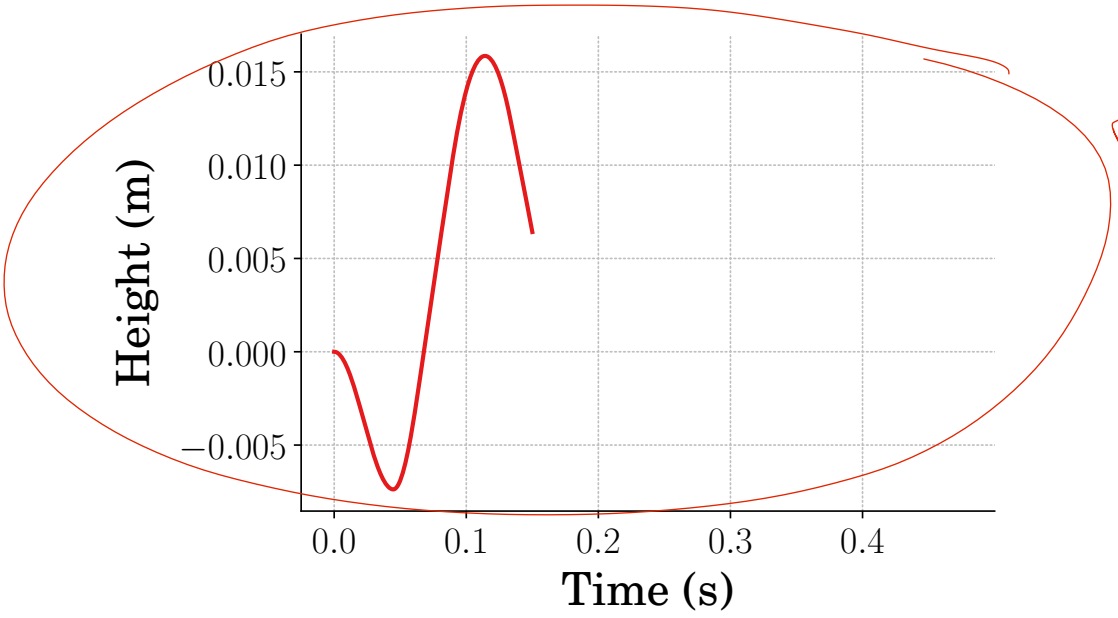


Figure 22. Jumping Response for Maximum Vibration Bang-bang Impulse Sequence

If the command learned by the agent in Figure 12b is analyzed directly, an

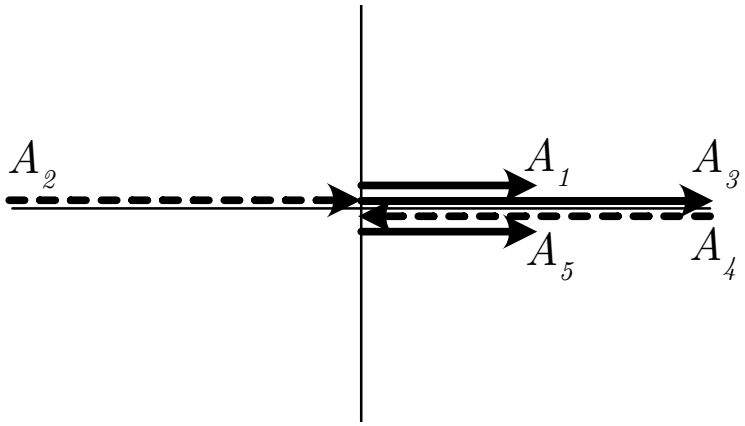


Figure 23. Vector Diagram for Learned Jumping Command

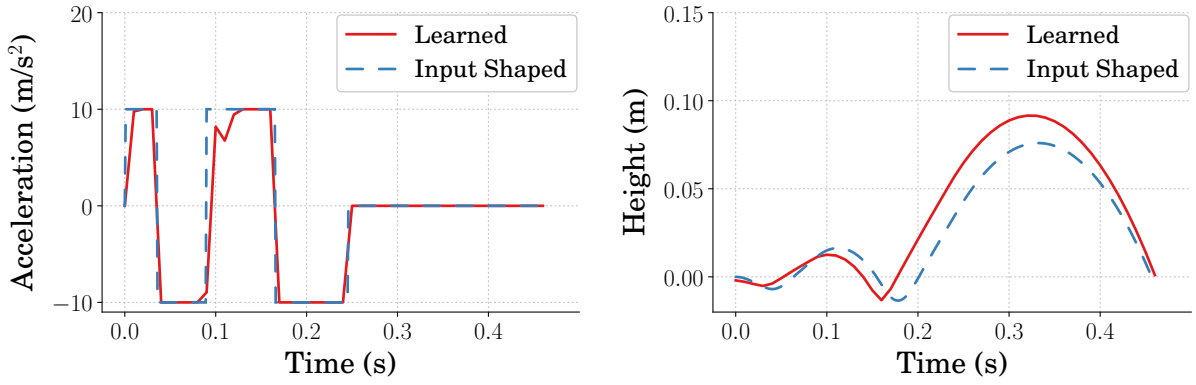
impulse sequence representing its bang-bang profiles can be found to be:

$$\begin{bmatrix} A_i \\ t_i \end{bmatrix} \approx \begin{bmatrix} 1 & -2 & 2 & -2 & 1 \\ 0 & \frac{\tau}{2} & \tau & 2\tau & 3\tau \end{bmatrix} \tag{18}$$

A vector diagram of this sequence of impulses is shown in Figure 23. From the vector diagram, the resultant vector does not reach the amplitude of the theoretical upper-bound case, but does result in vibration that is 400% of that which a single, unity magnitude impulse would create.

The learned command and the one resulting from the convolution of this impulse sequence are shown in Figure 24a. The timing and amplitude of the command generated using the impulse sequence are a good match for the learned policy. This is further confirmed by comparing the jumping responses for the two commands, which are shown in Figure 24b.

In Figure 24b, the learned controller achieves a slightly greater jump height than the input-shaped approximation of it, but the general trends between the two match. The disparity in height is likely do to the input-shaped command being defined for a linear approximation of the monopode jumping system, whereas the RL-policy was trained on the the nonlinear system. Both commands where evaluated on the nonlinear system. Regardless of the disparities, the results confirms that the methods presented in this chapter are a feasible option for analyzing control strategies learned using RL



(a) Jumping Commands **(b)** Jumping Responses

Figure 24. Comparison of Learned Controller and Input Shaping Approximation techniques.

3.4 Conclusion

An approach to design jumping commands using input shaping was reviewed. Then, the best learned stutter jumping command from Chapter 2 was analyzed using methods enabled approximating the command with an input-shaped step command. This analysis showed that the RL-learned command does not directly match the theoretical maximum for a linearized model of the monopode system. However, both the commands and responses for the shaped approximation and the RL agent matched closely, suggesting that input shaping is a feasible option for the analysis of learned policies.