Toys, Entertainment robots, Videos Games: Challenges in Design and Control

Pranav Bhounsule

Department of Mechanical Engineering
University of Texas at San Antonio

March 23, 2018

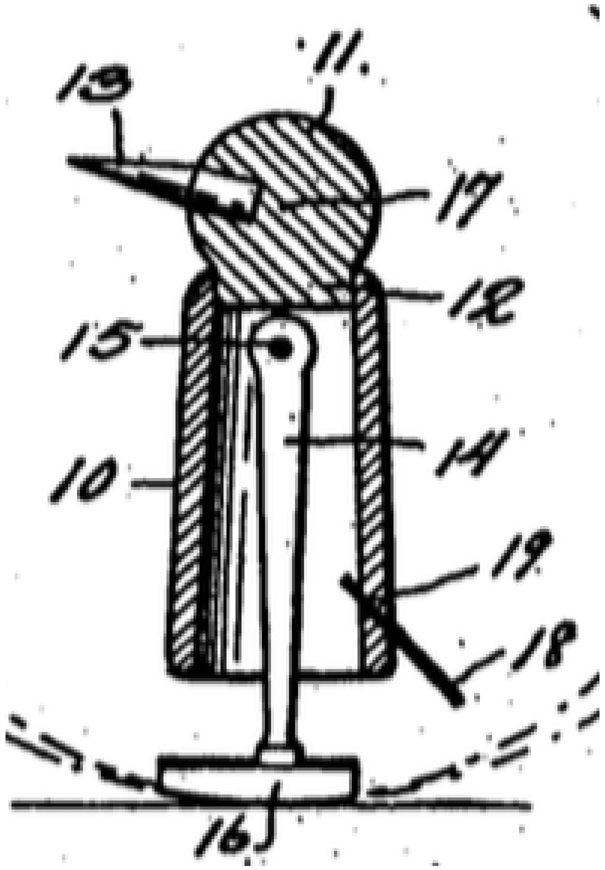
(1) Toys that walk/run

Walking robot: Honda Asimo

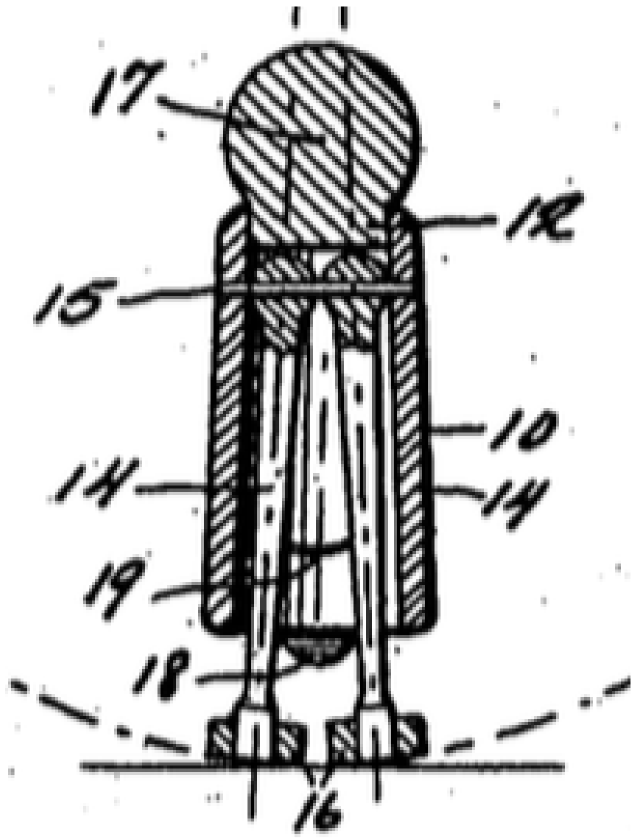
Wilson Walker

Wilson walker

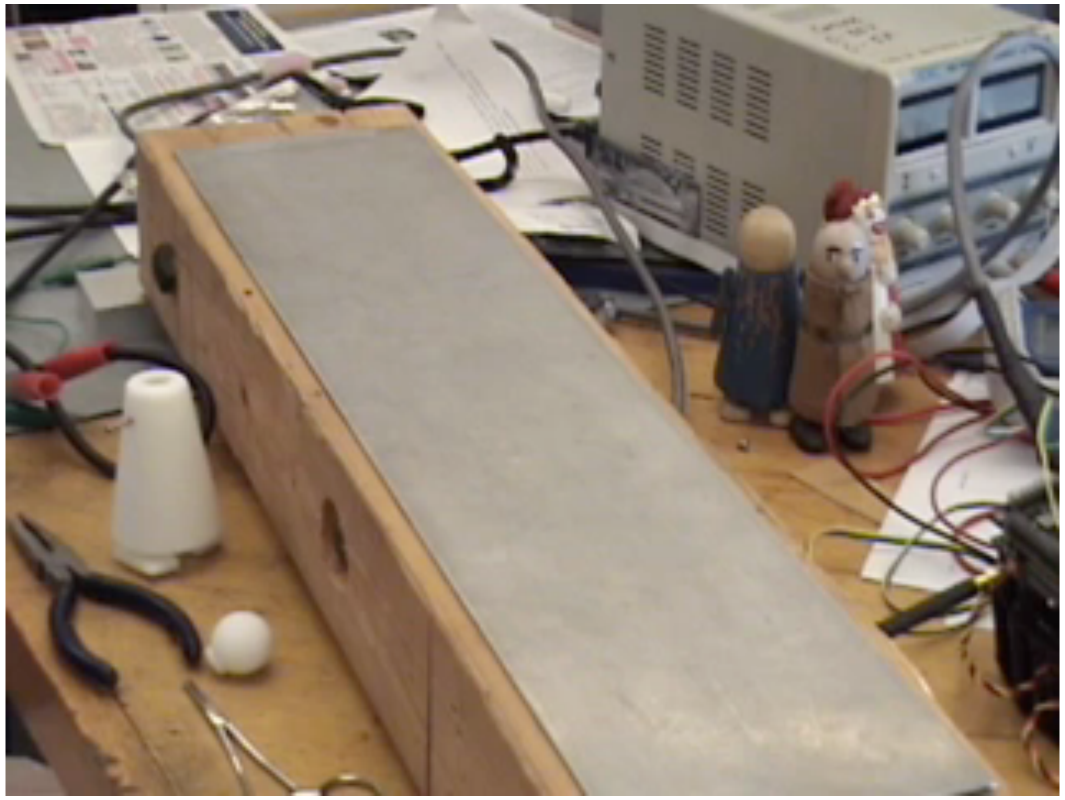
(A)



(B)

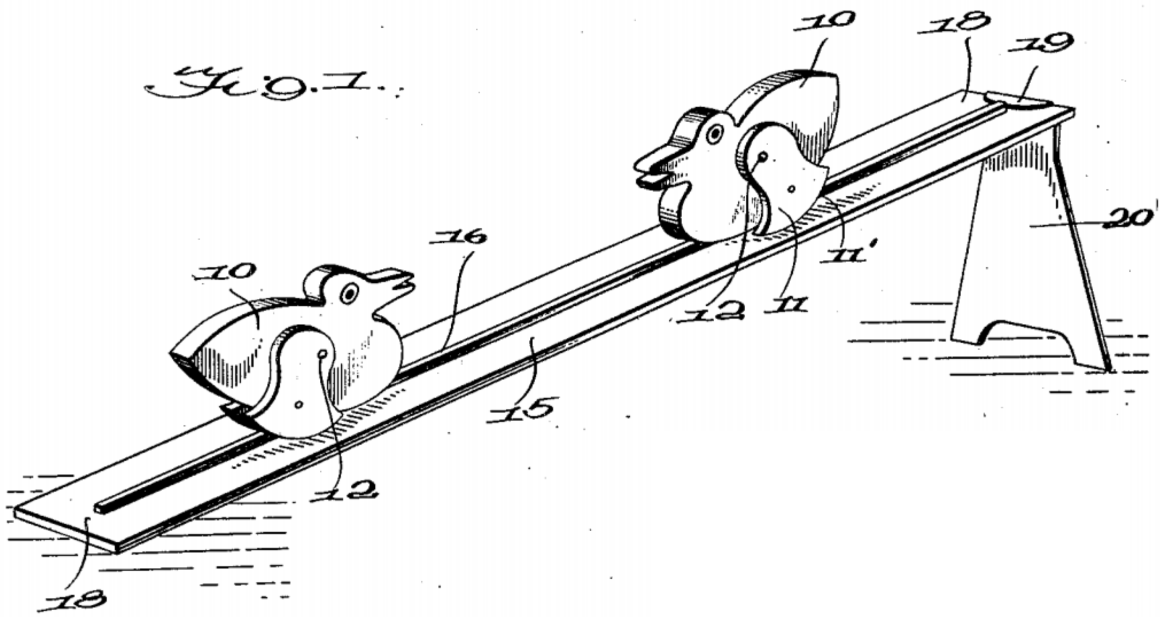


(C)

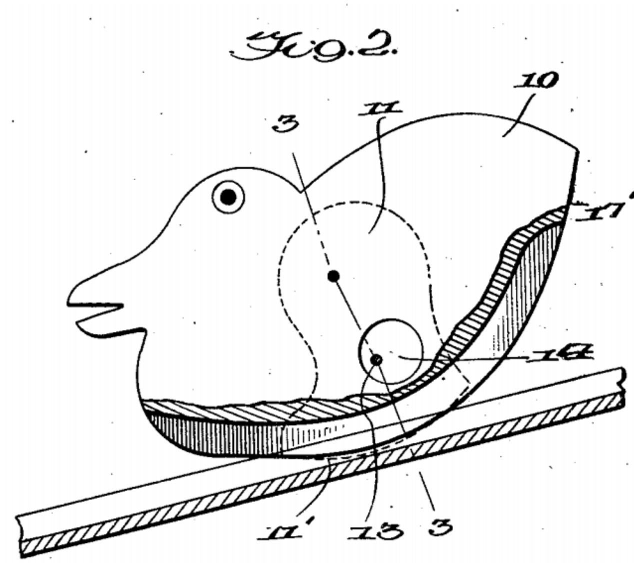


Ravert patent

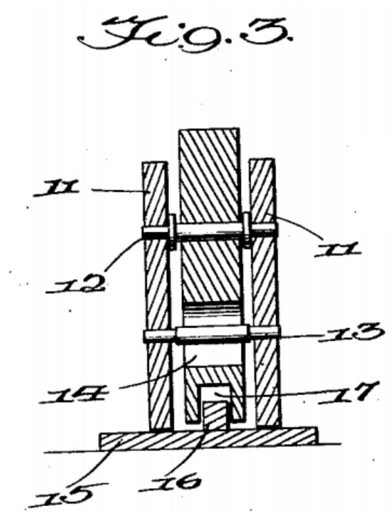
(A)



(B)



(C)

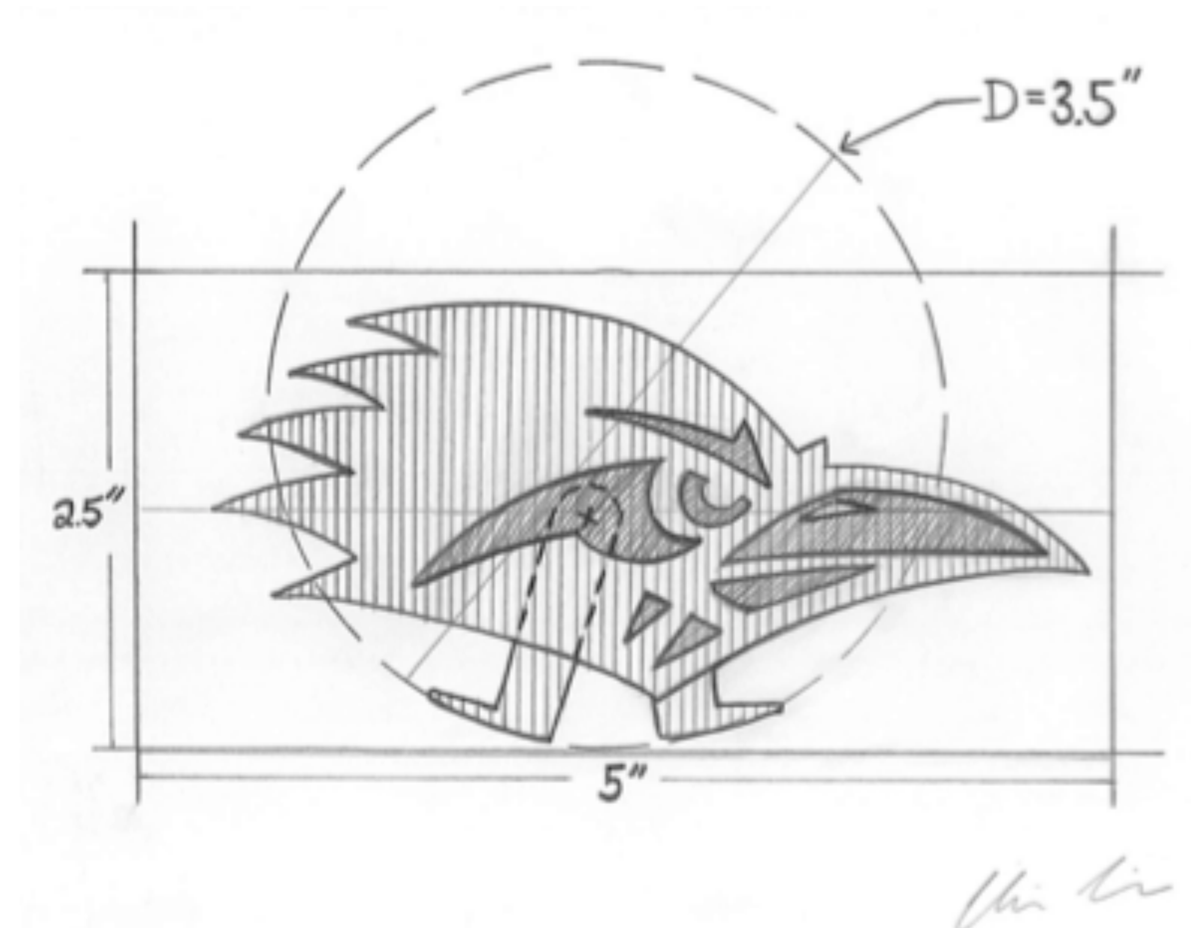


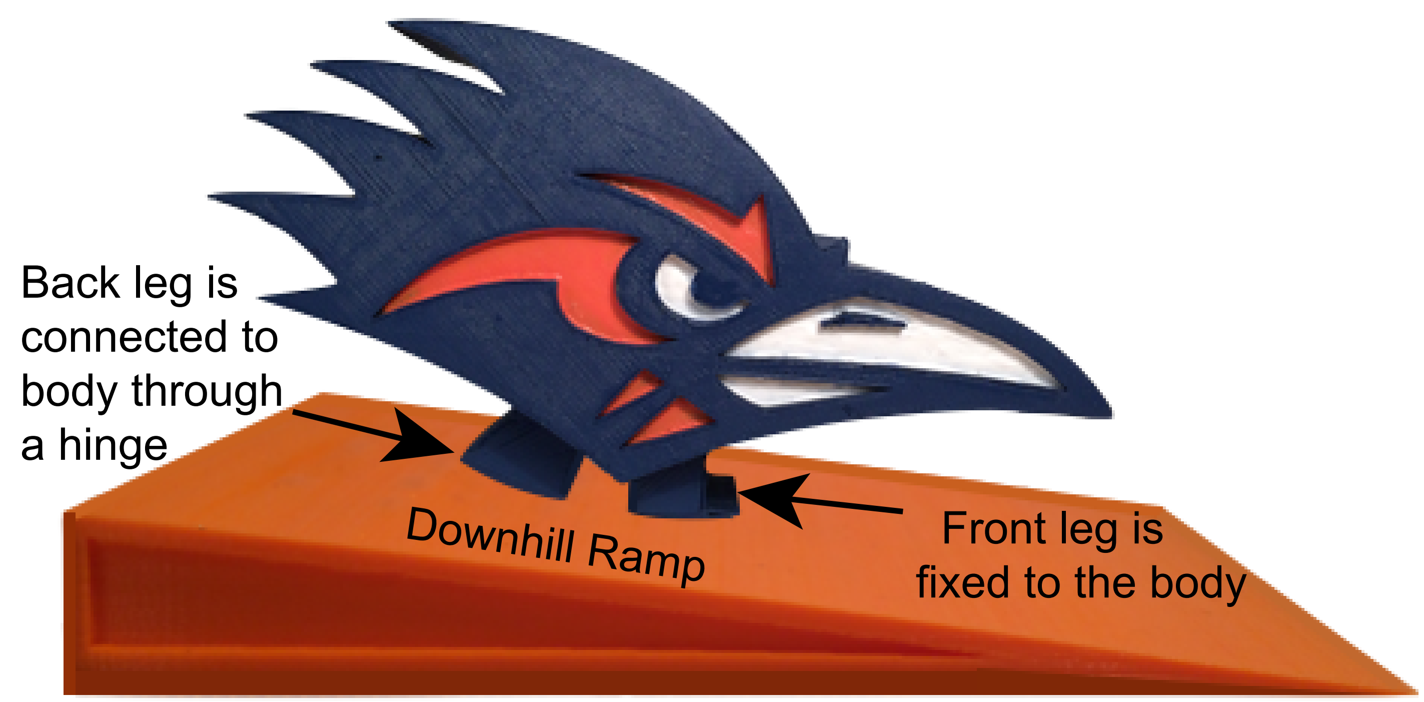
UTSA mascot, Rowdy

Rowdy Walker

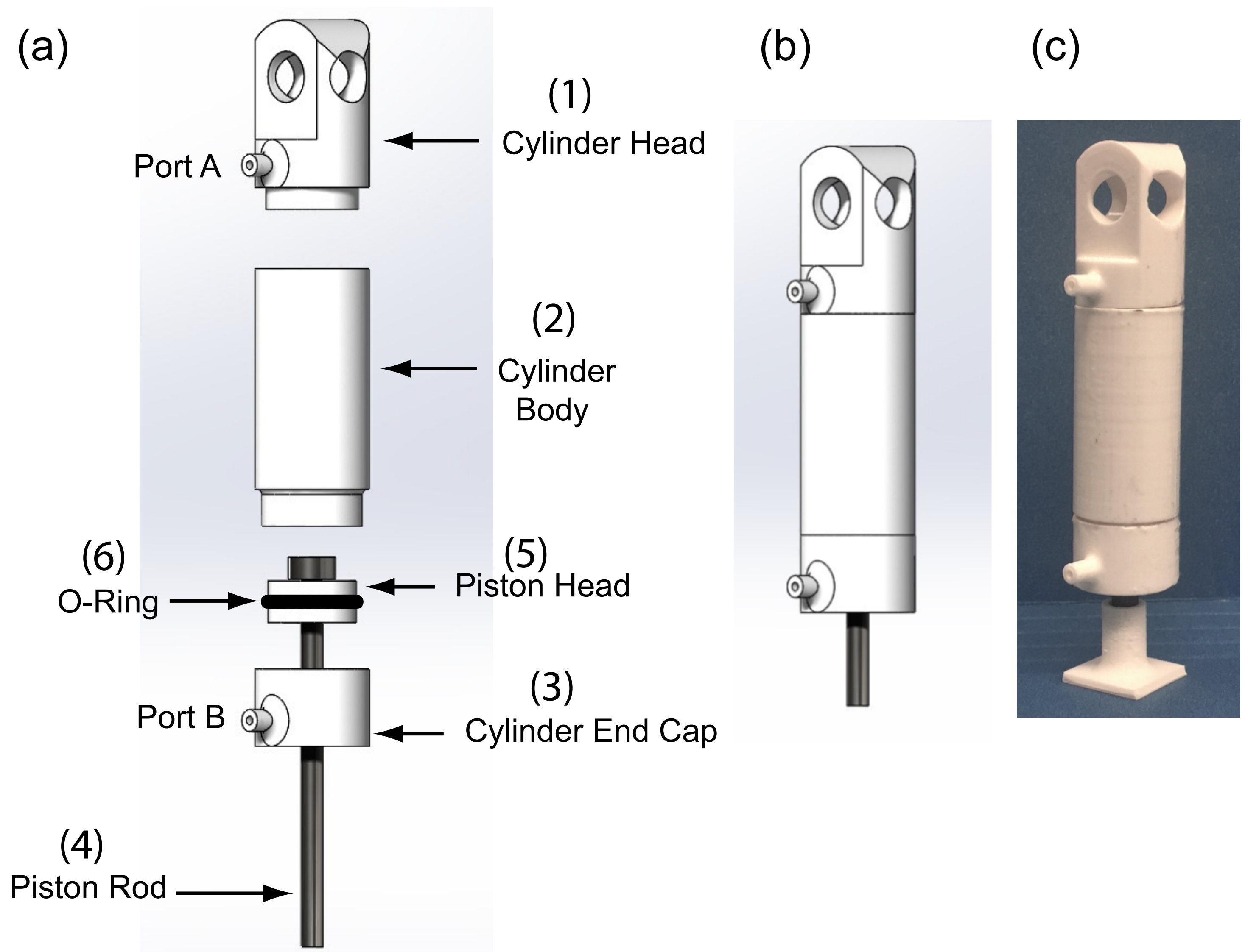
Methods & Challenges

- Leg Design
- Mass Distribution
- Integrated Hinge
- Support Material
- Commercialization?
    - Time
    - Cost

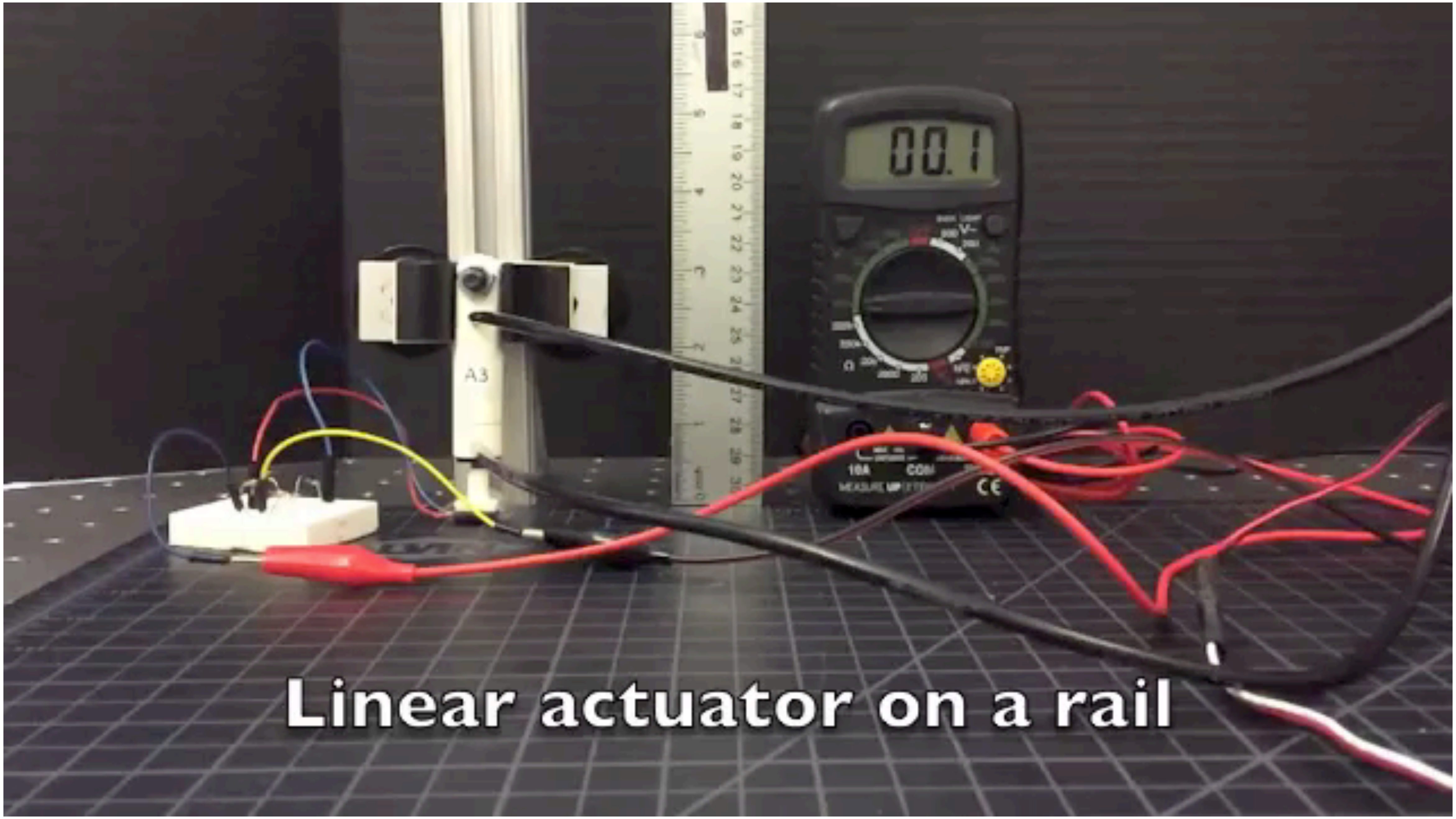


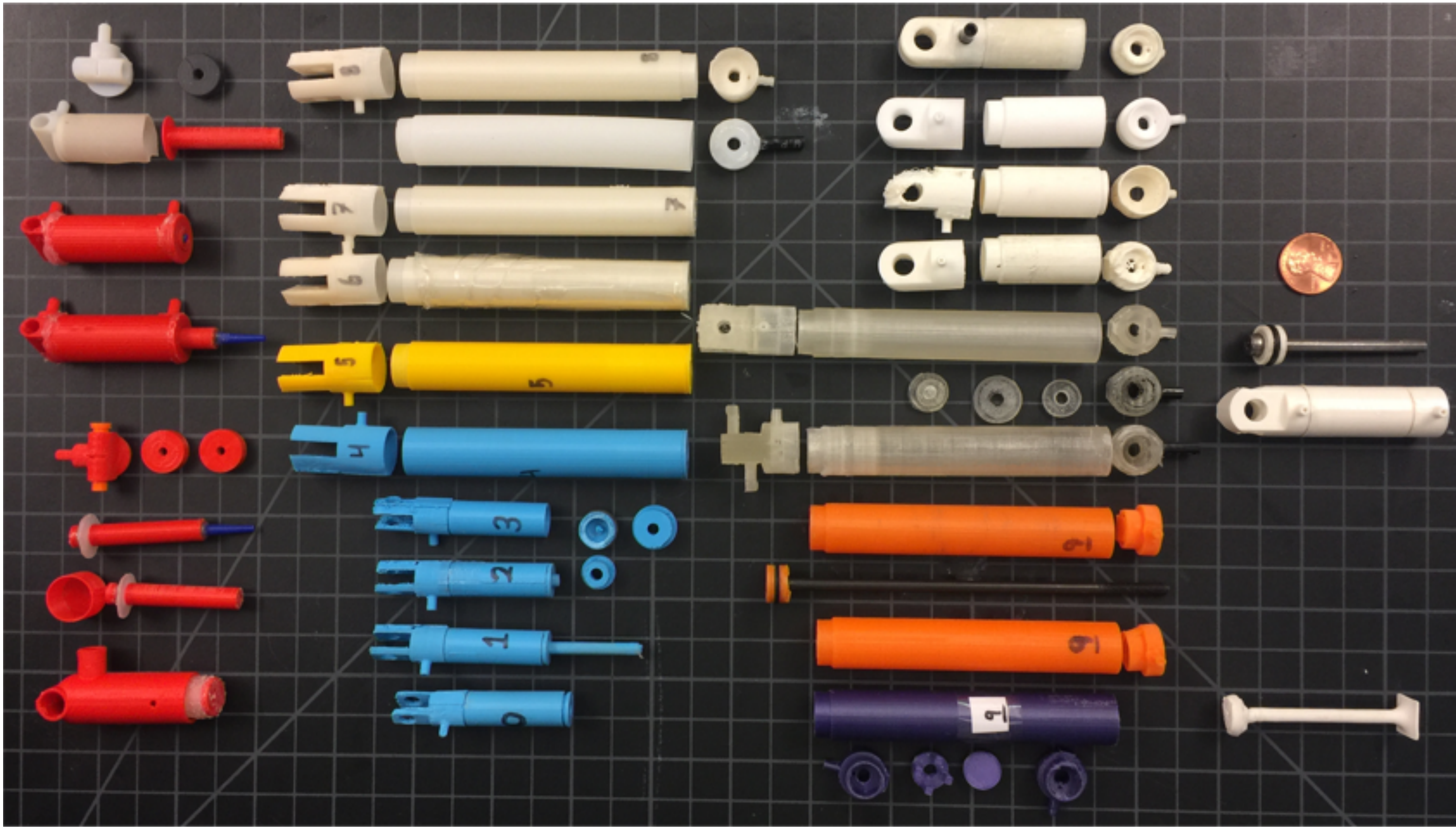


3D printed, linear, ON-OFF, pneumatic actuator



Actuator working





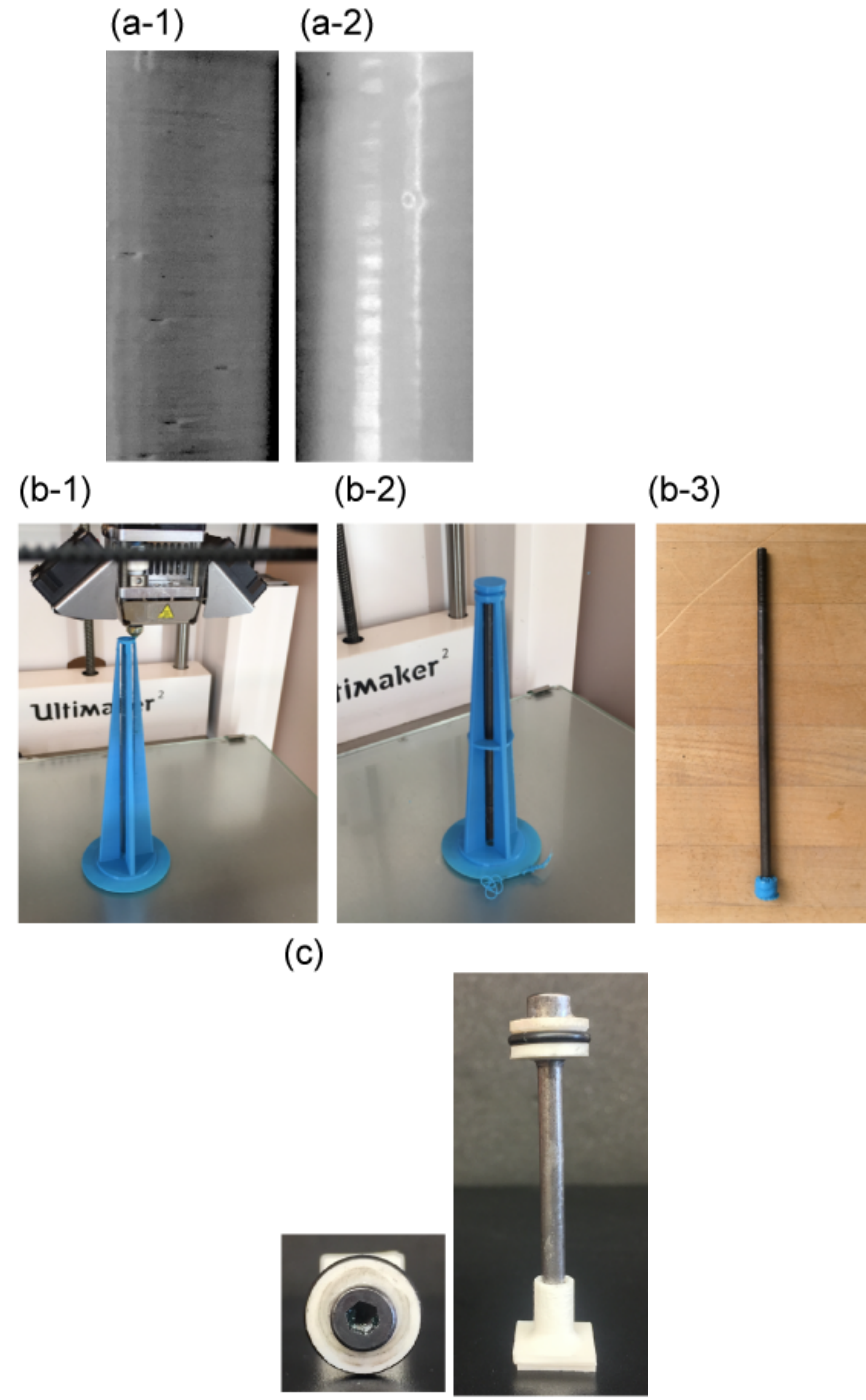
Methods & Challenges

(a) Pores (Acetone)

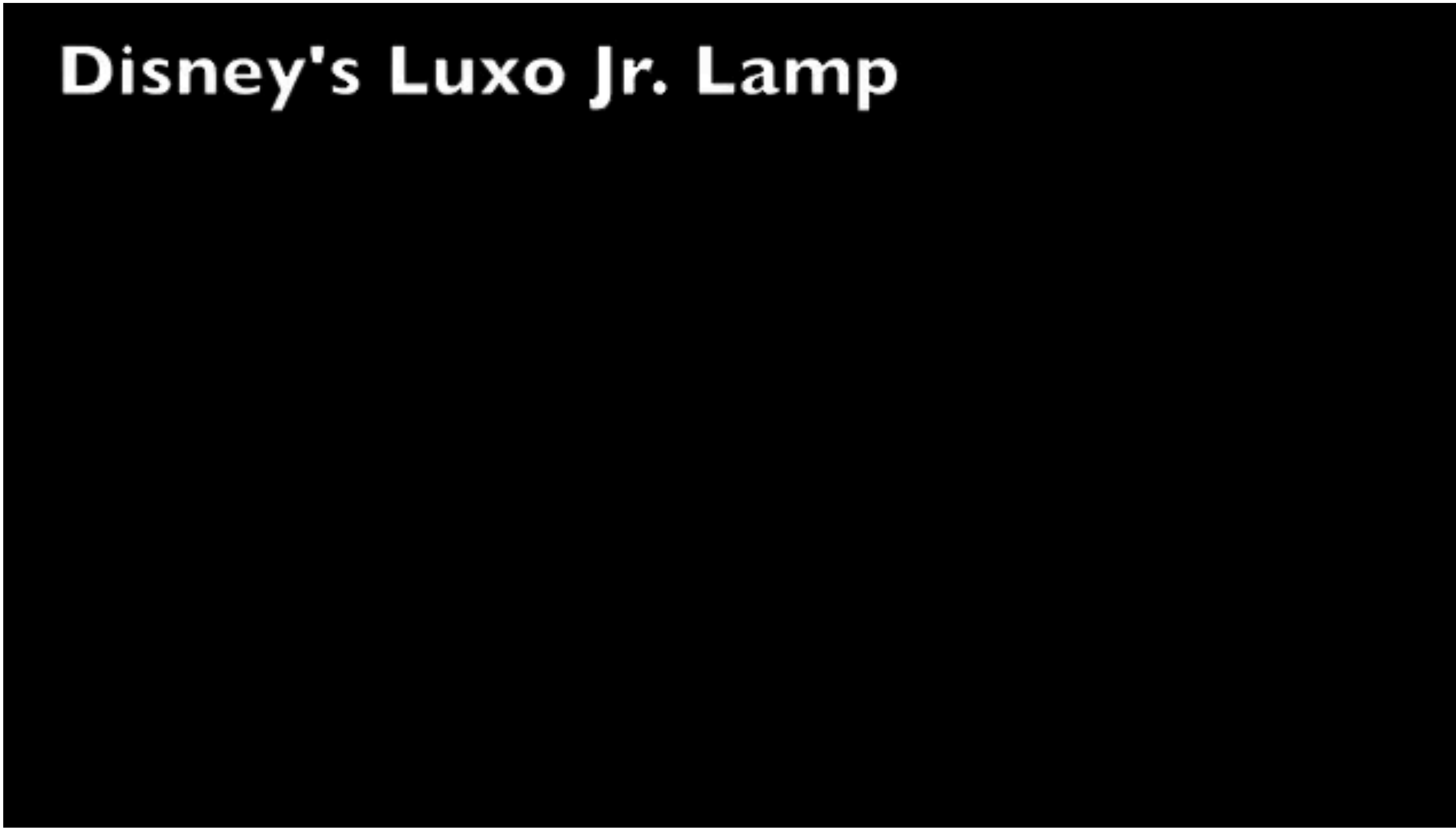
(b) Strength (Embedding)

(c) Piston - Cylinder interface

- Viton O-rings
- Waterproof greese



Disney's Luxo Jr. Lamp

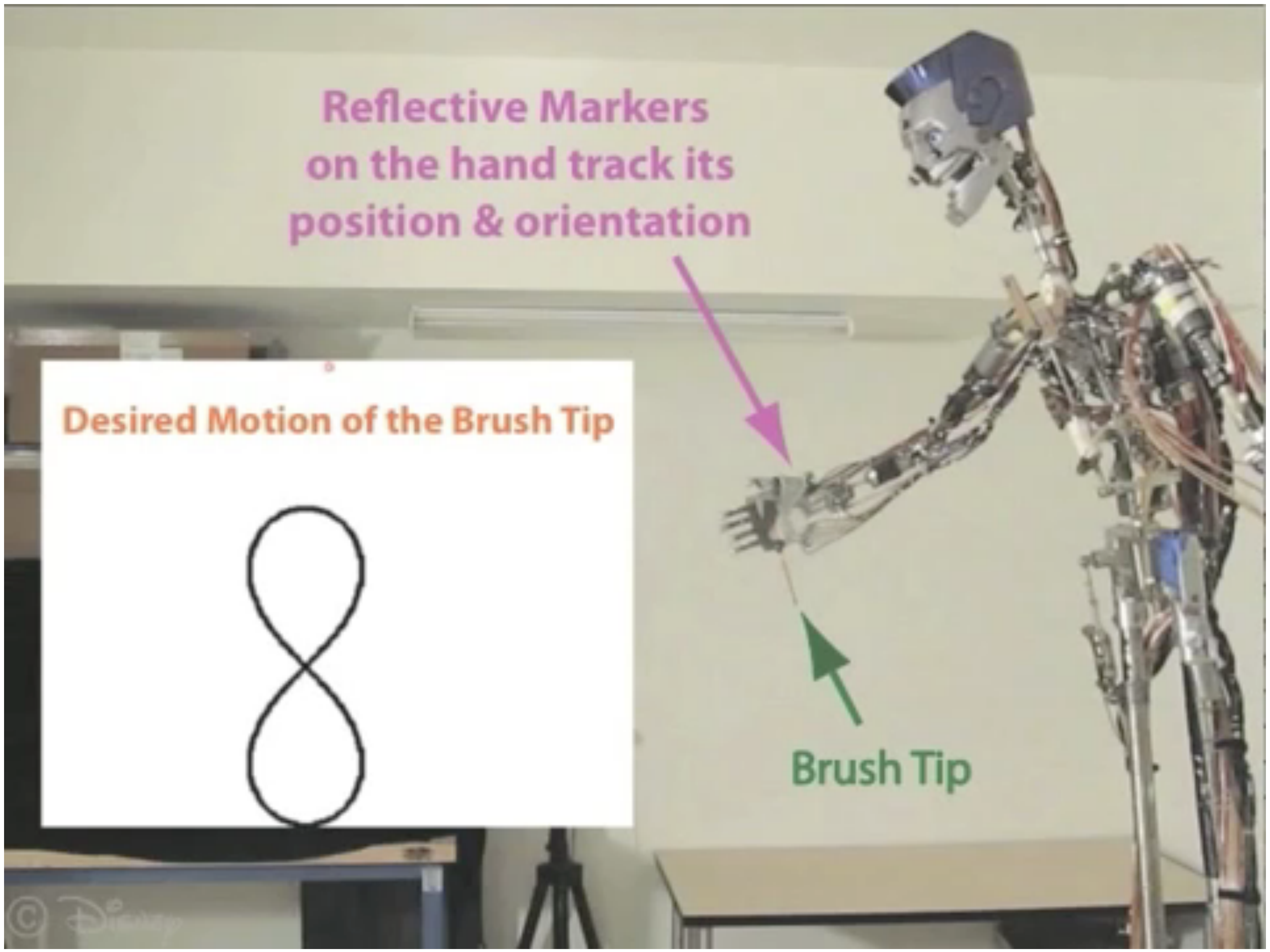


(2) Entertainment Robots

Disney animatronics

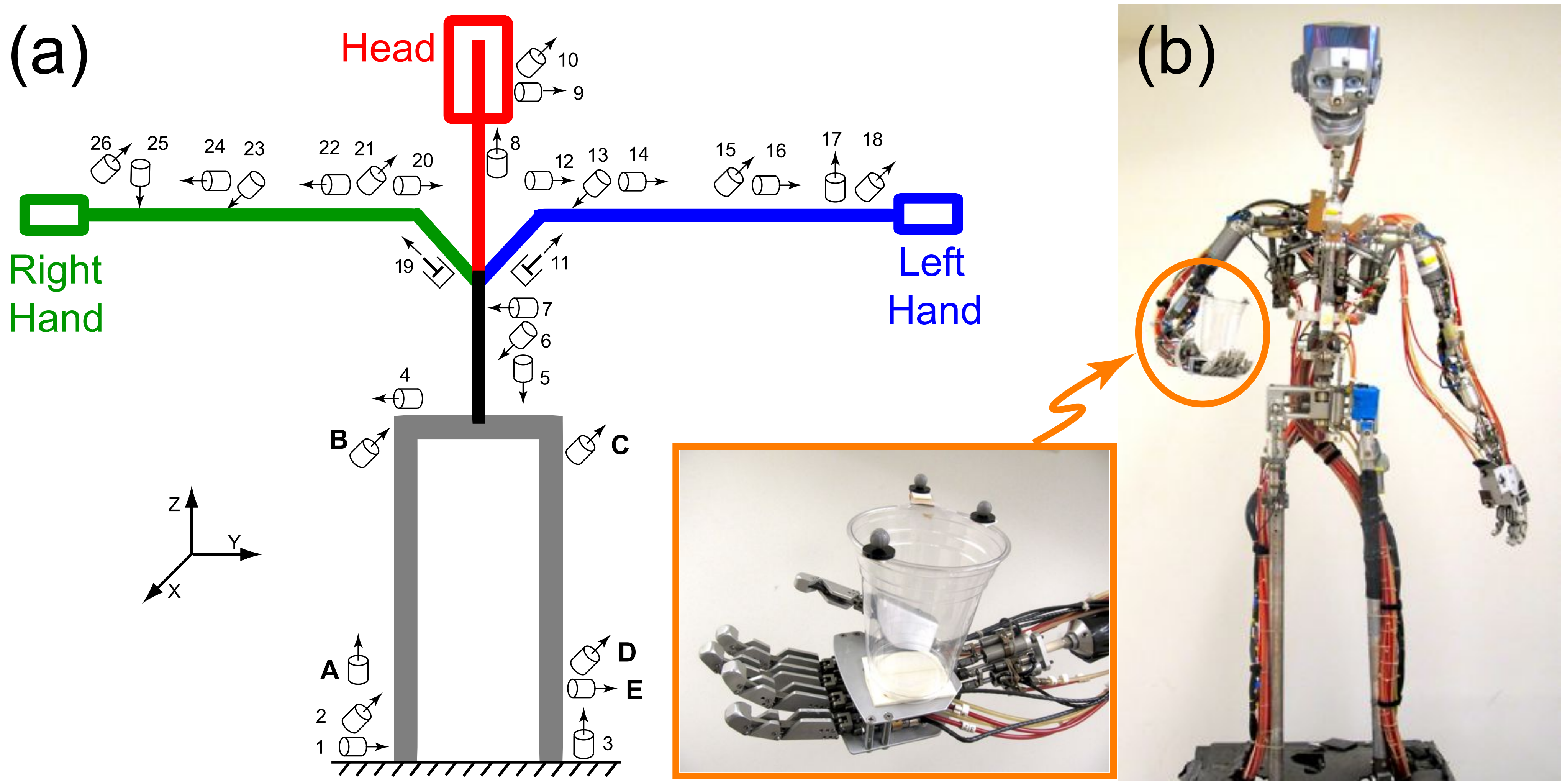
- Manually tuned
- Time consuming

Inverse kinematics



- Bhounsule & Yamane, Humanoids 2015

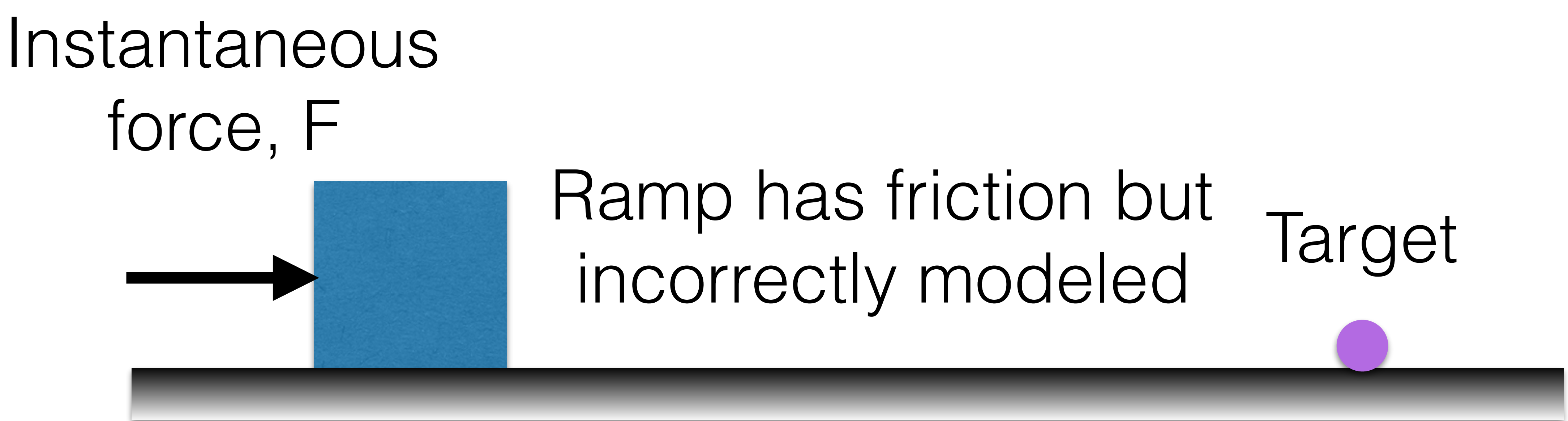
Issues with Kinematics model



- Flexible joints —-> Rigid body models are invalid
- Low bandwidth control —> poor servo operation
- High degrees of freedom —> Error magnification
- Wear and Tear —> Part/link replacement

Iterative Learning Control (ILC): 1-D example

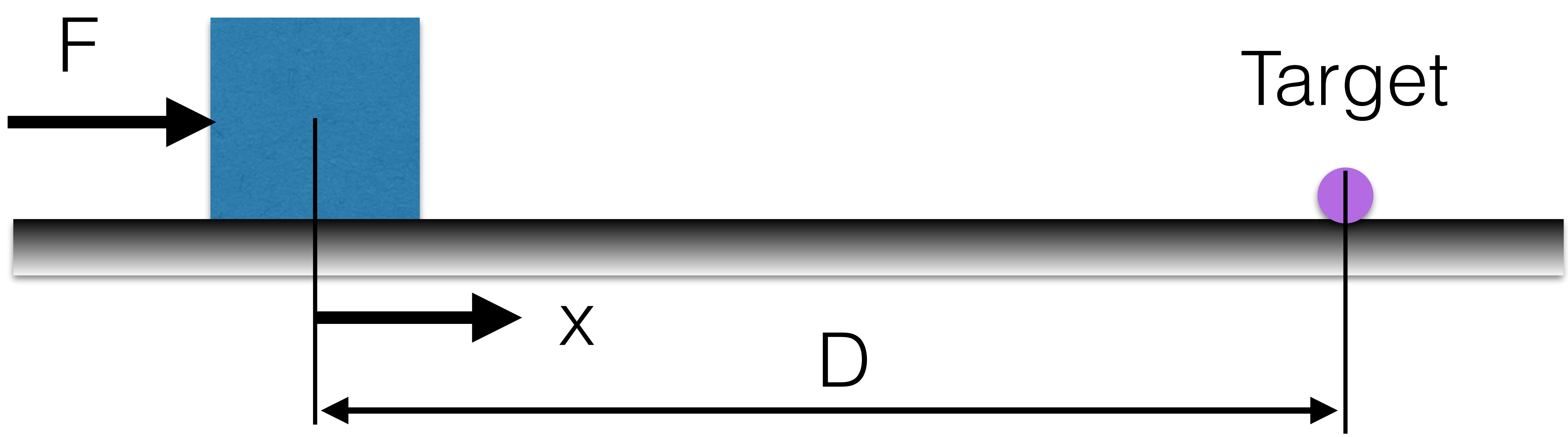
Problem: Move block to the target by applying an instantaneous force



Iterative Learning Control (ILC): 1-D example

Model: $$x = f(F, \mu)$$

Imprecise

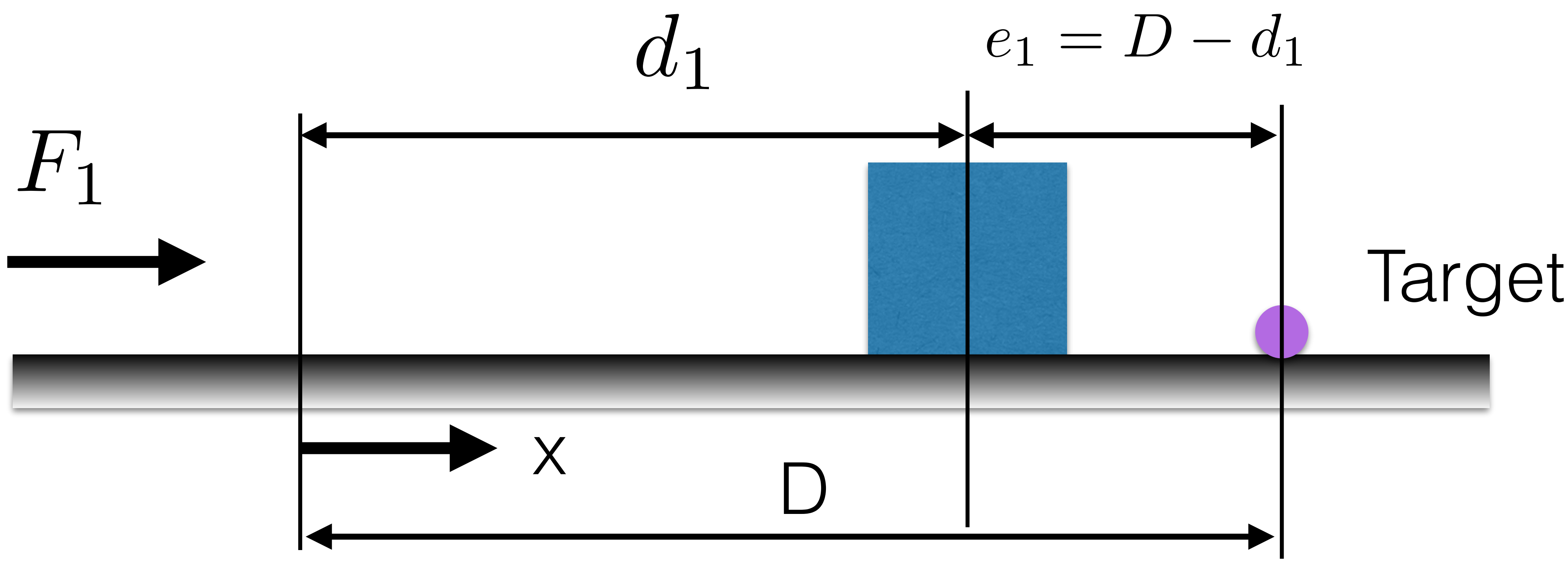


Inverse: $$F = f^{-1}(x, \mu)$$

Imprecise

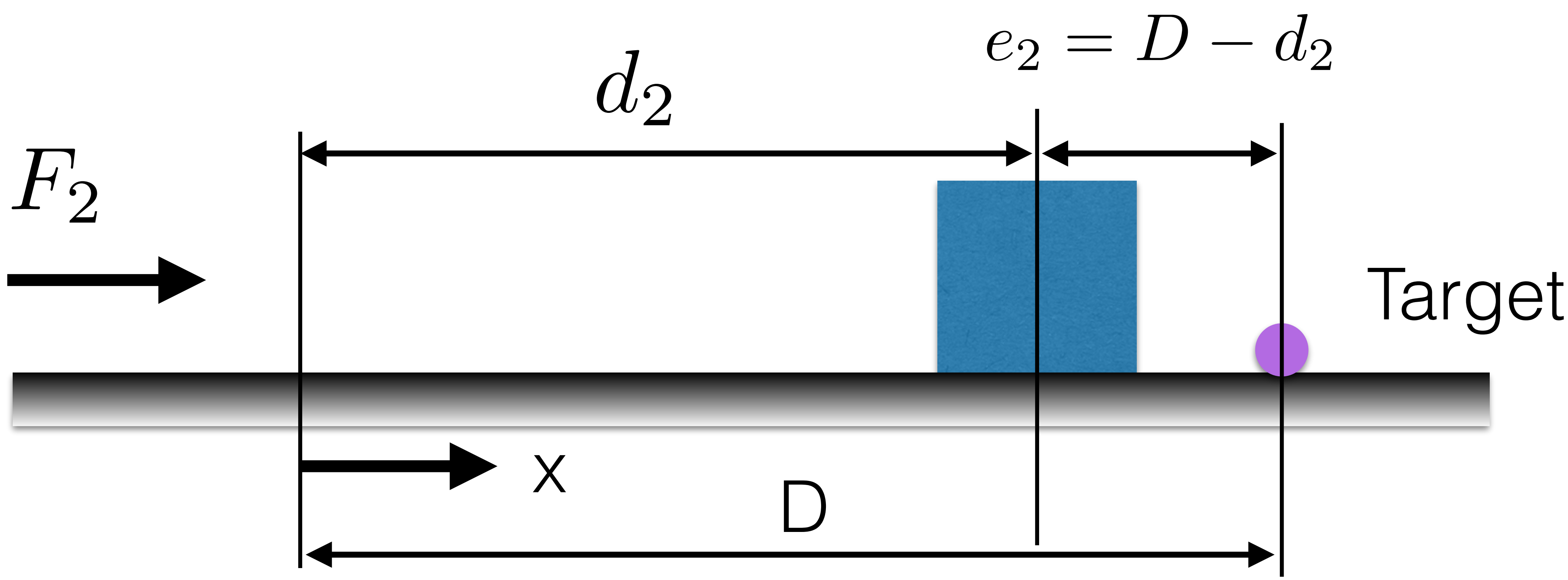
1-D example (trial 1)

Control (trial 1): $F_1 = f^{-1}(x, \mu)$



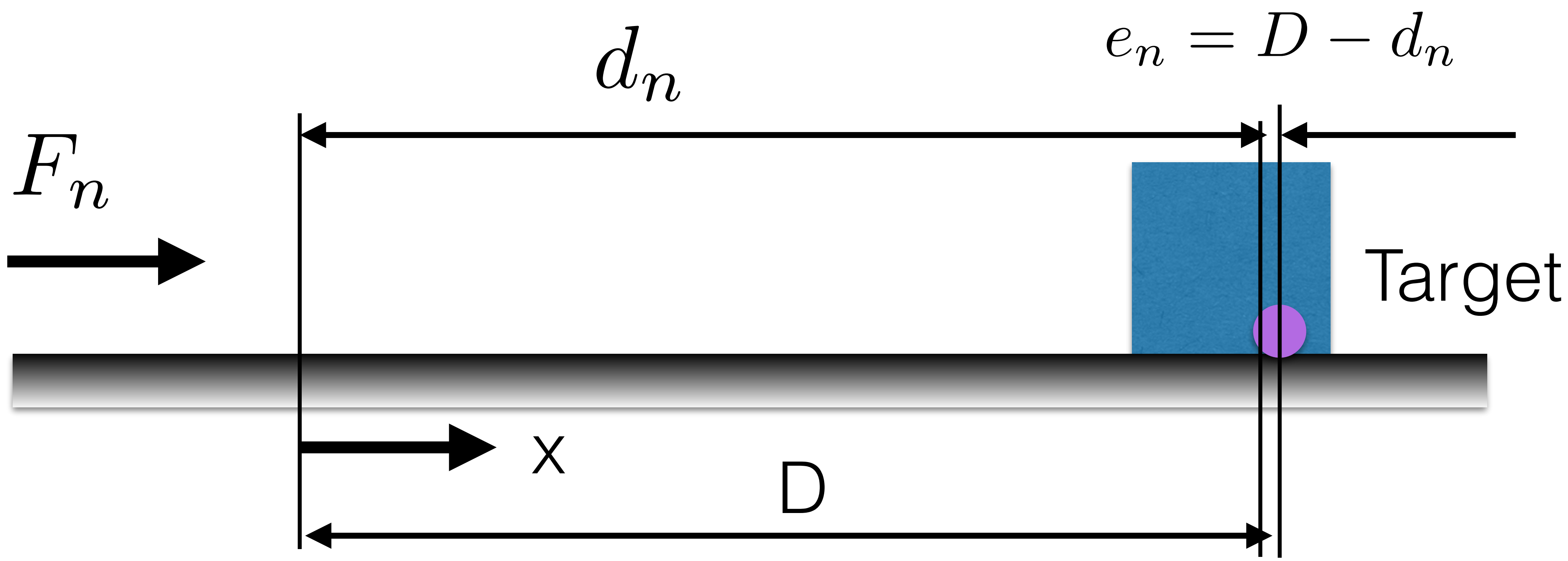
1-D example (trial 2)

Control (trial 2): $F_2 = F_1 + \lambda e_1$



1-D example (trial 2)

Converged when e_n is small



Our approach: Non-linear Inverse Kinematics (IK) update

$$\mathbf{Y}_{\text{des}}^{i+1} = \mathbf{Y}_{\text{des}}^{i} + \gamma(\mathbf{Y}_{\text{ref}} - \mathbf{Y}^{i}),$$
$$\boldsymbol{\theta}^{i+1} = \hat{\mathbf{F}}^{-1}(\mathbf{Y}_{\text{des}}^{i+1}),$$

where:

θ^i	Angle command trial i
Y_i	End-effector position trial i
Y_{ref}	End-effector reference
Y_{des}^i	Desired end-effector for IK
$\hat{F}$	Estimated Forward Kinematics Model
γ	Learning gain

Our approach: Non-linear Inverse Kinematics (IK) update

$$\mathbf{Y}_{\text{des}}^{i+1} = \mathbf{Y}_{\text{des}}^{i} + \gamma(\mathbf{Y}_{\text{ref}} - \mathbf{Y}^{i}),$$

$$\boxed{\theta^{i+1} = \hat{\mathbf{F}}^{-1}(\mathbf{Y}_{\text{des}}^{i+1}),}$$

Find non-linear IK within joint limits

Inverse Kinematics computation

$$\boxed{\boldsymbol{\theta}^{i+1} = \hat{\mathbf{F}}^{-1}(\mathbf{Y}_{\mathrm{des}}^{i+1}),}$$

Use nonlinear constraint optimization for IK

$$g(\boldsymbol{\theta}) = \sum_{i=1}^{n_{\mathrm{dof}}} (\theta_i - \theta_i^{\mathrm{rest}})^2,$$

Cost: Bias toward a pose

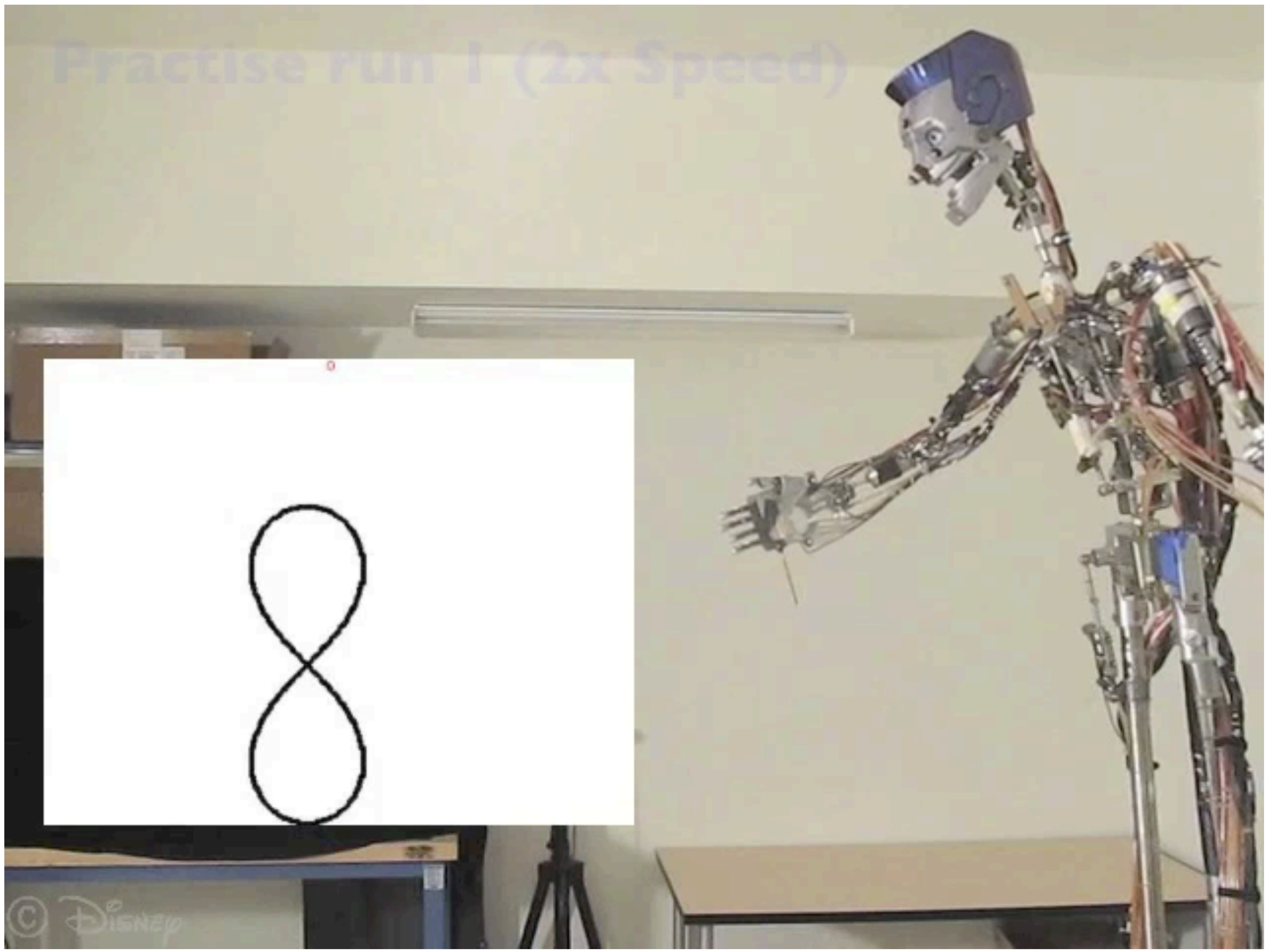
$$h_1(\boldsymbol{\theta}) = x_{\mathrm{des}} - x_{\mathrm{ref}} = 0,$$
$$h_2(\boldsymbol{\theta}) = y_{\mathrm{des}} - y_{\mathrm{ref}} = 0,$$

End-effector constraint: Satisfy estimated end-effector position

$$h_3(\boldsymbol{\theta}) = \boldsymbol{\theta} - \boldsymbol{\theta}_{\min} \geq 0,$$
$$h_4(\boldsymbol{\theta}) = \boldsymbol{\theta} - \boldsymbol{\theta}_{\max} \leq 0.$$

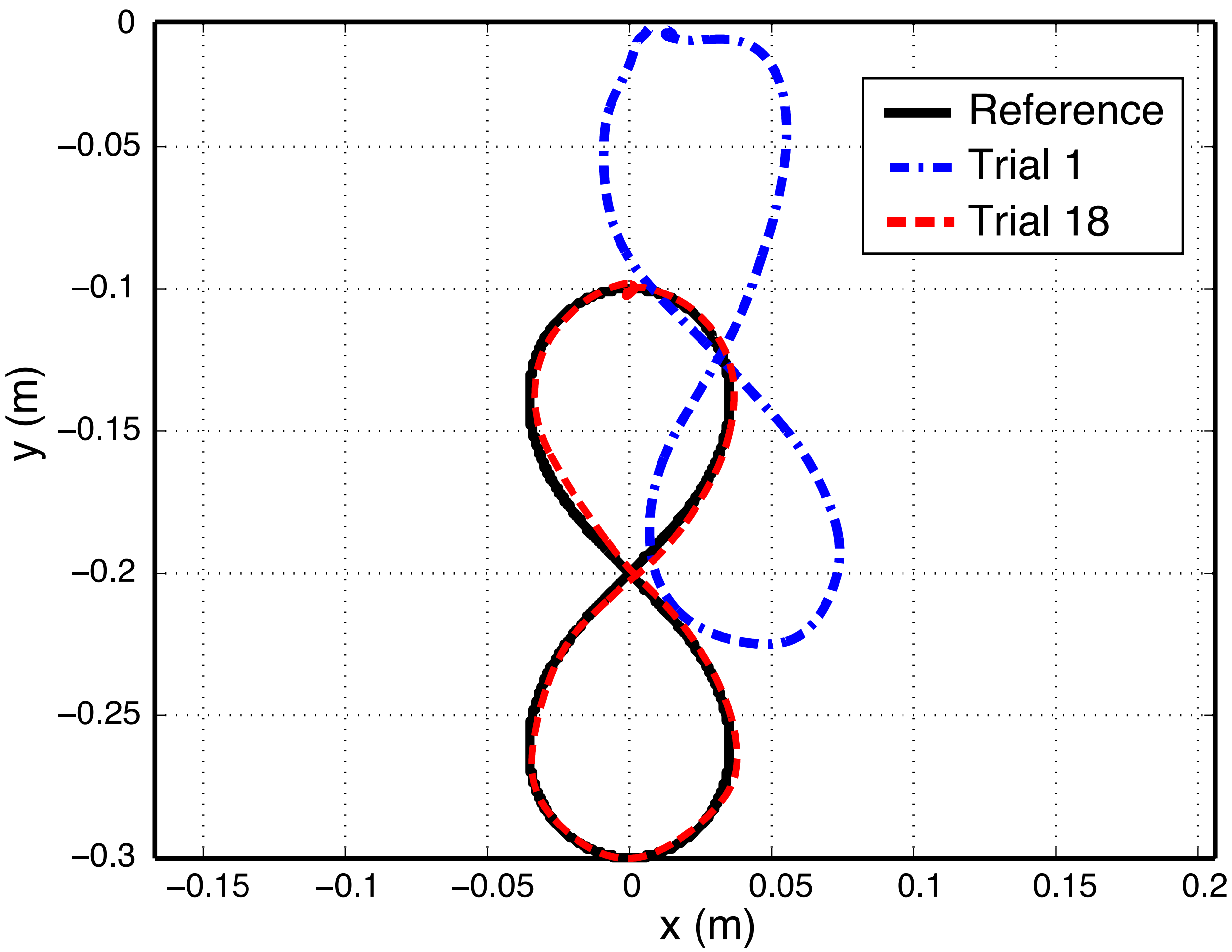
Joint constraint: Satisfy joint limits

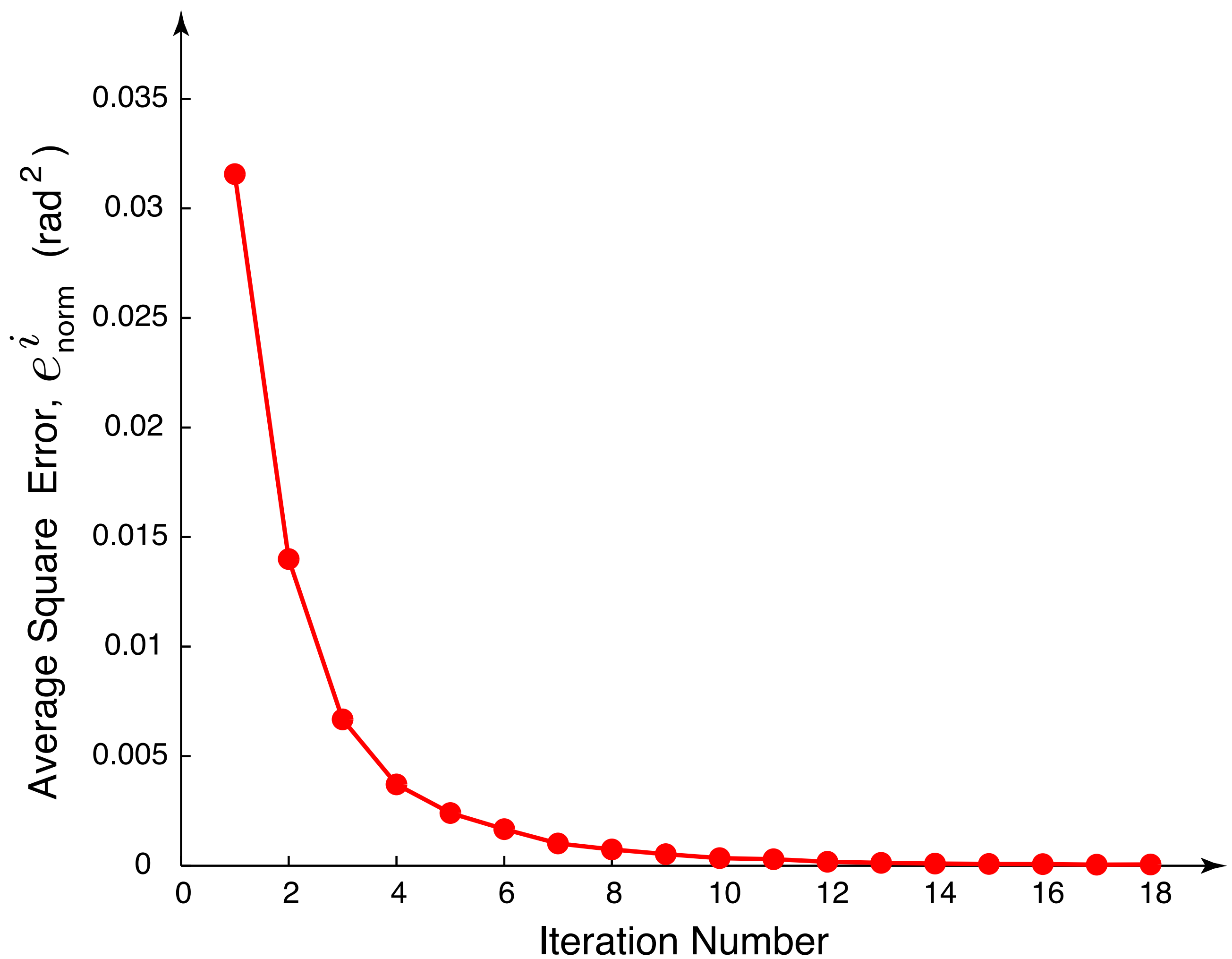
Inverse kinematics with Iterative Learning Control

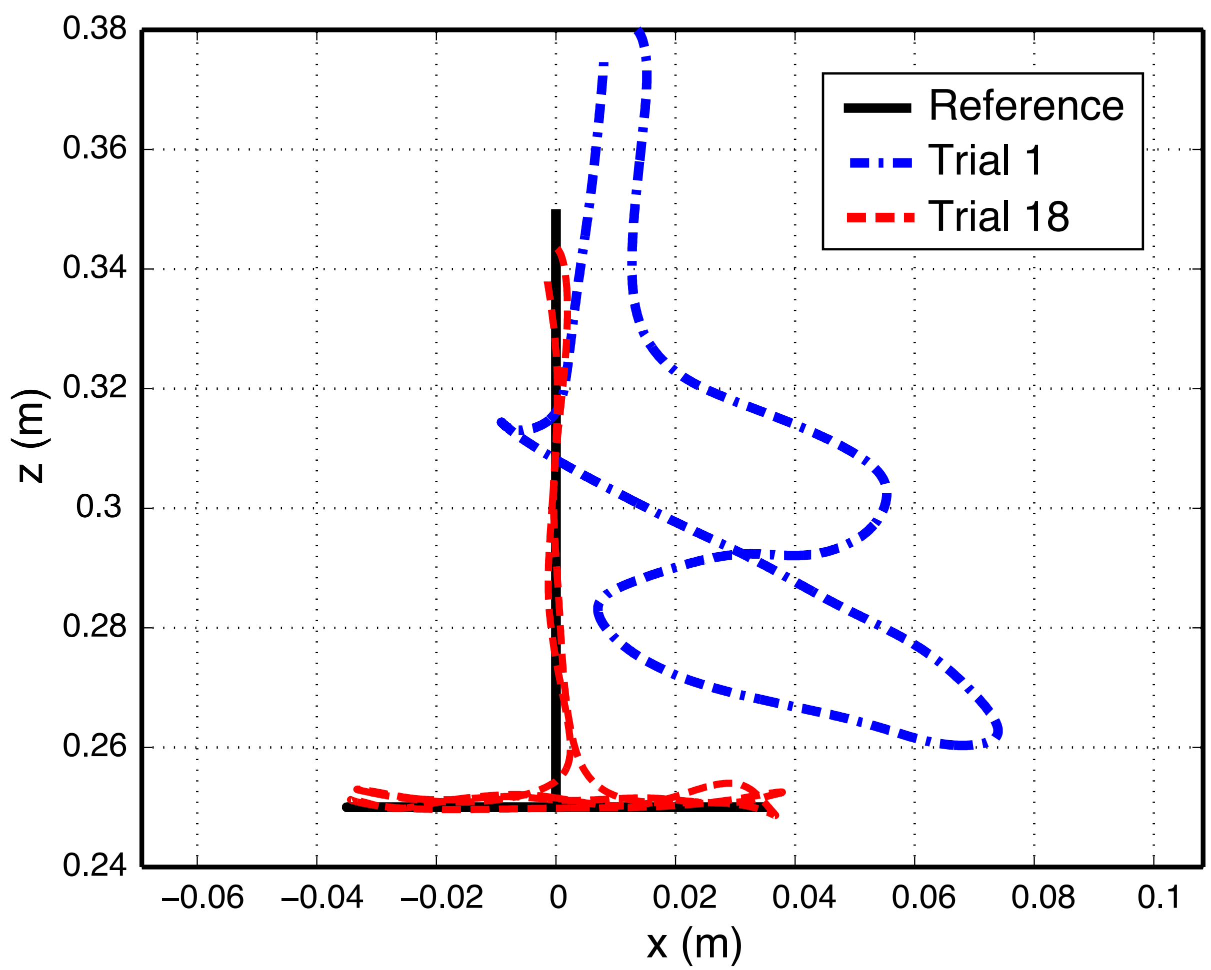


- Bhounsule & Yamane, Humanoids 2015

Results for writing task

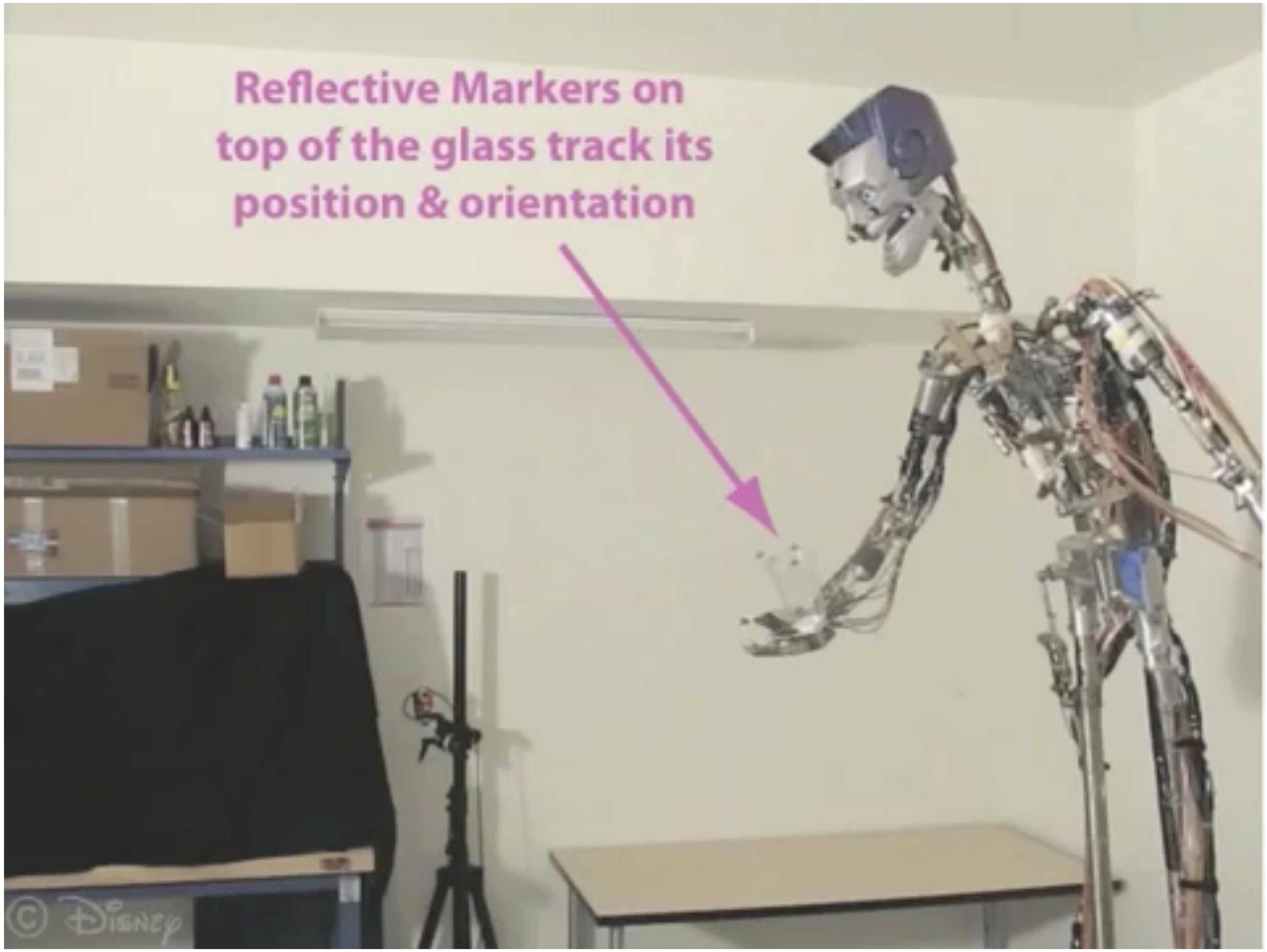






- Convergence: 18 trials
- Trial 1: Error ~ 1e-3
- Trial 18: Error ~ 1e-5

Other tasks



- Bhounsule & Yamane, IJHR 2017

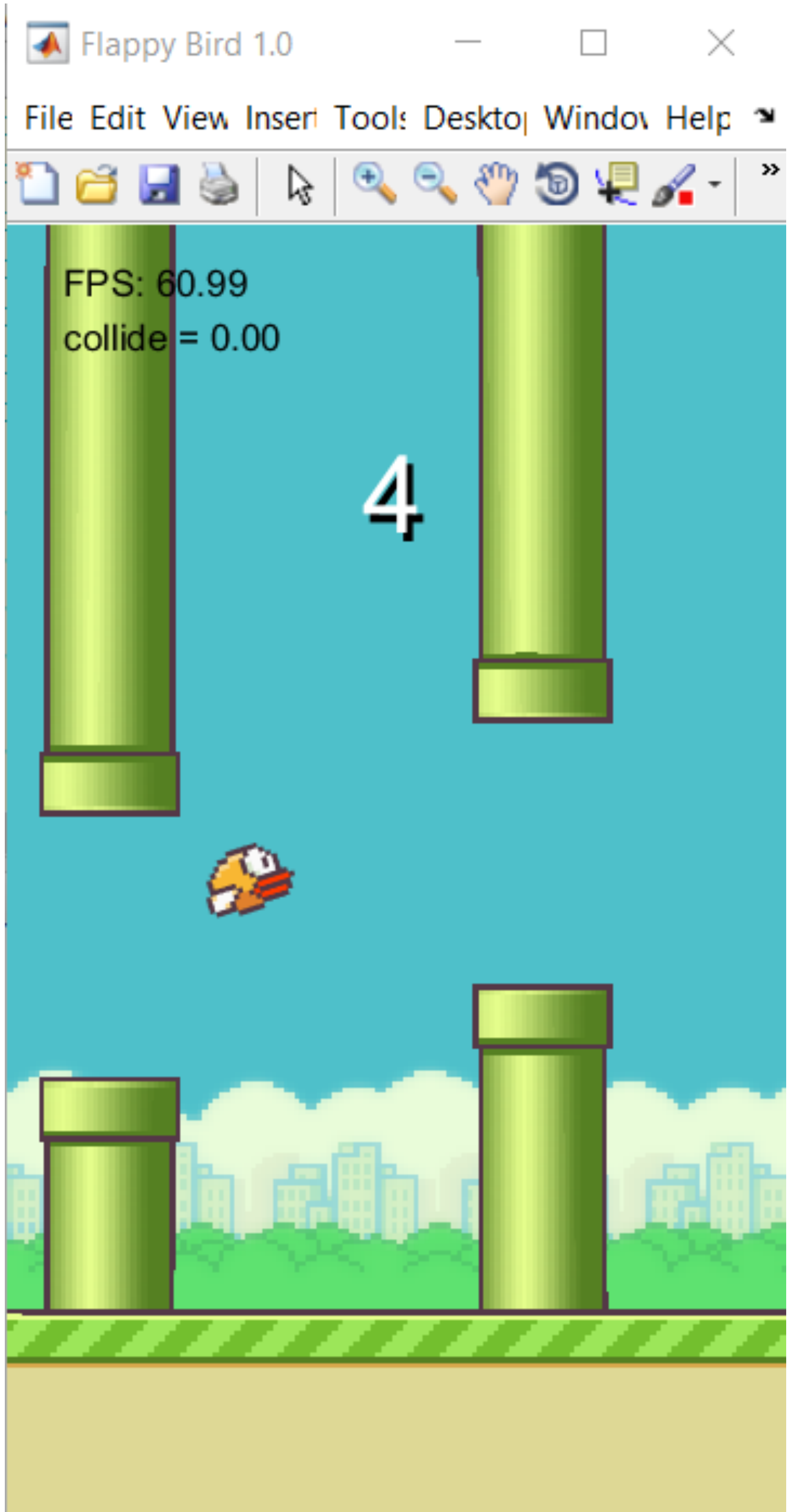
(3) Video Games

Flappy Bird Game (iPhone/android)

Control: Tap screen to navigate bird through pipes

Scoring: 1 point/pipe passed

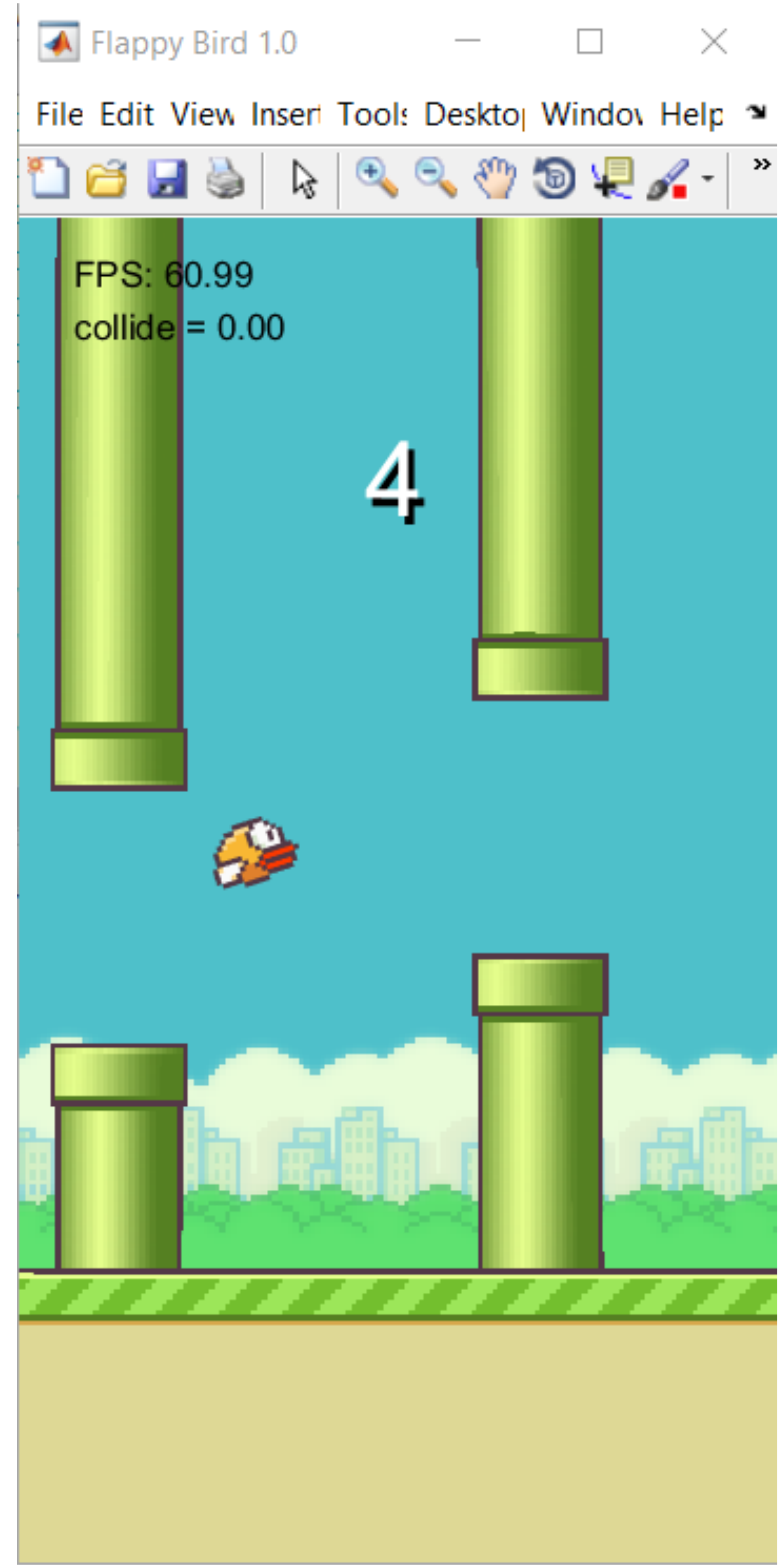
Objective: Maximize points.



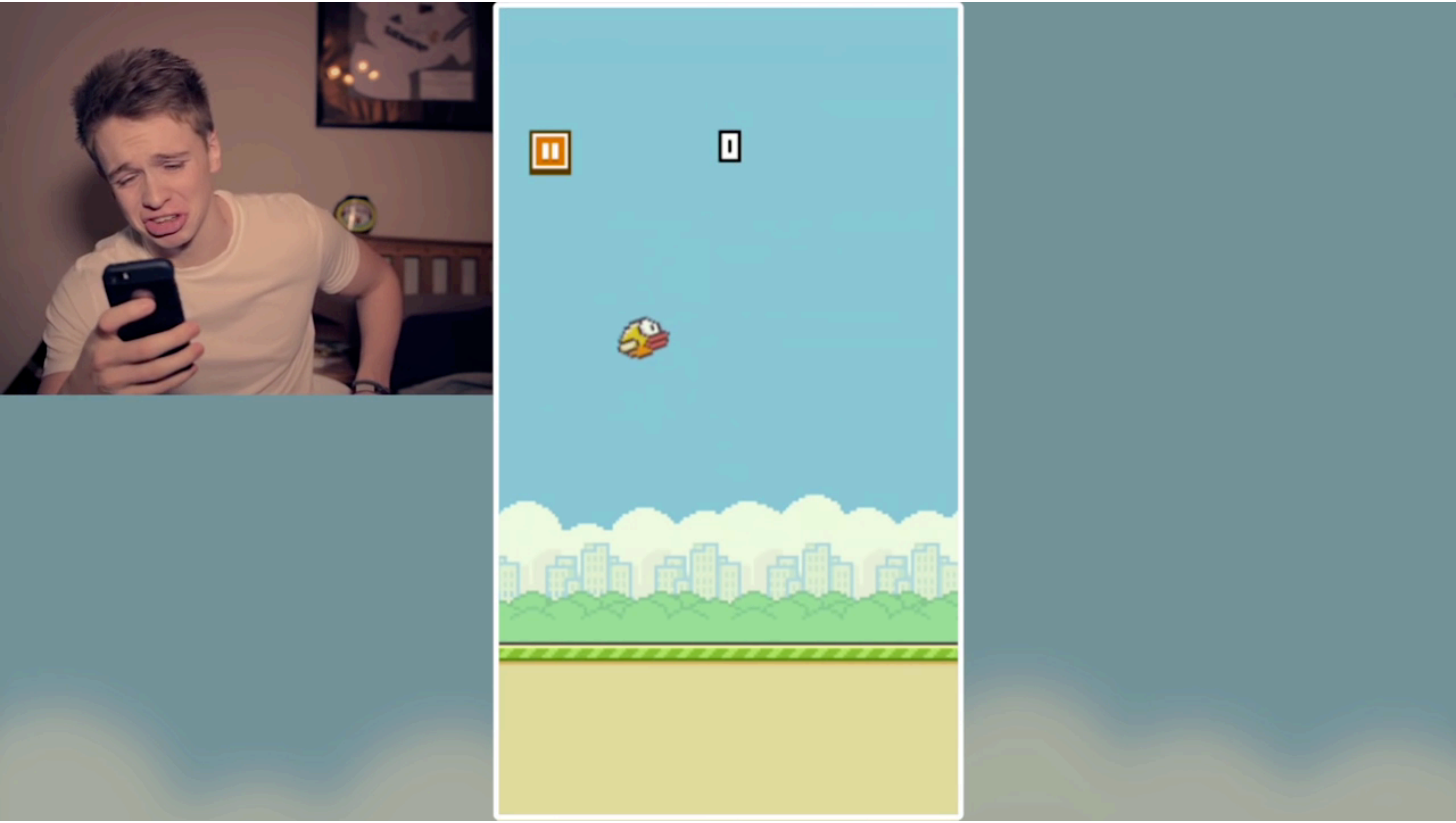
Flappy Bird Game

History:

- May 2013: Game released
- Jan 2014: Most downloaded game on iTunes, earning $50,000/day (?)
- Feb 2014: Game removed from iTunes by developer citing its addictive nature



Flappy Bird, simple concept but difficult to achieve high score



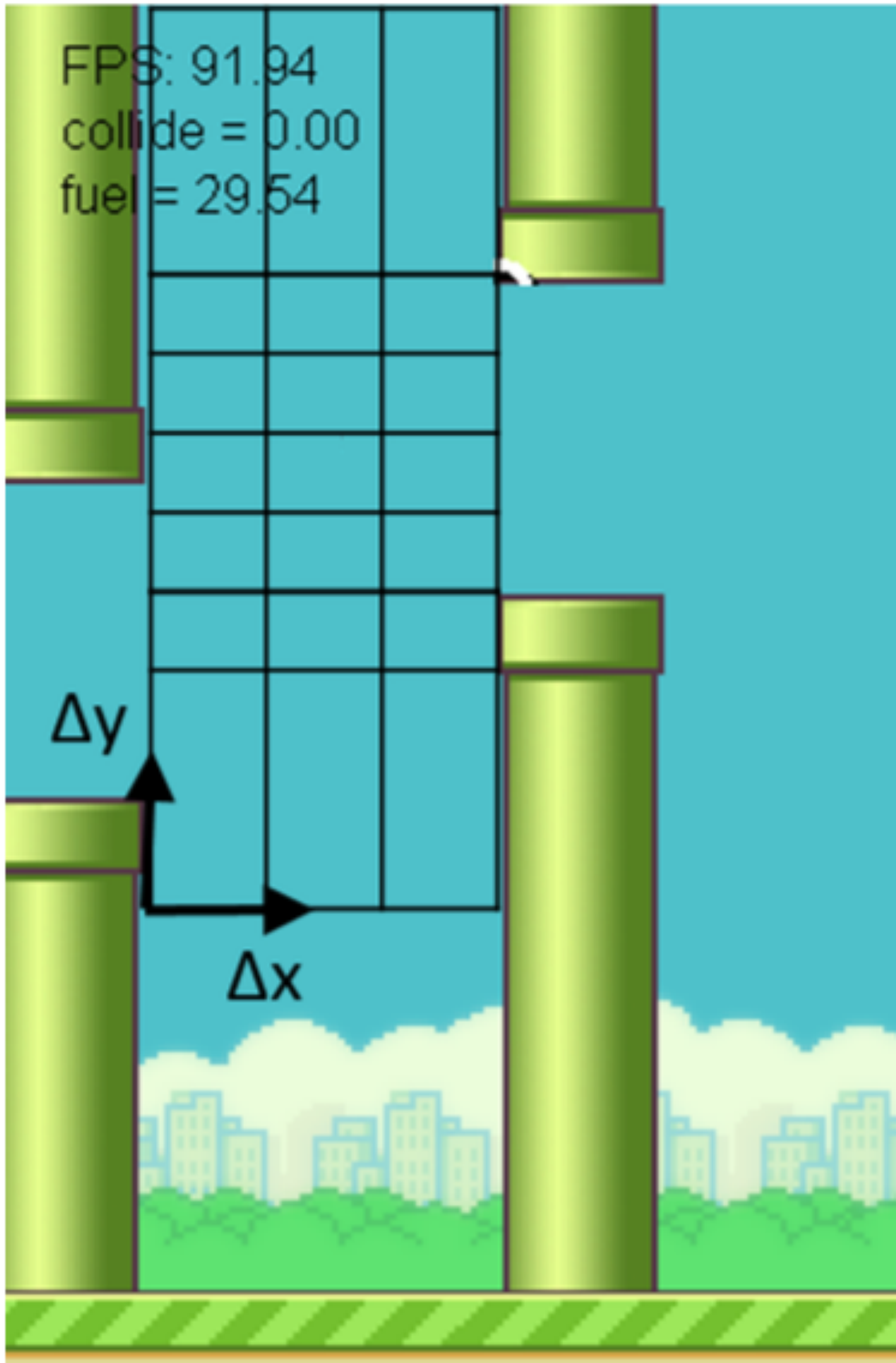
How to beat Flappy Bird

downloaded from a YouTuber

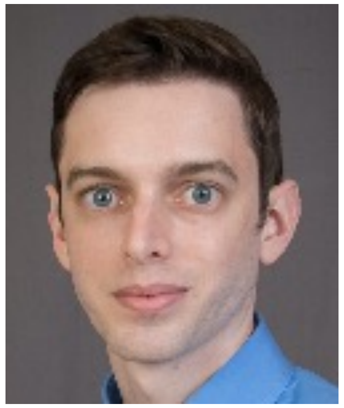
Past work

Machine Learning

- Reinforcement learning,
- Q-learning, and
- Support Vector Machines.
- Select features,
- Learn state-action pairs
- Scores ~100-1500



Discretization
(In dimension Δx, Δy)



Physics

$$Y_{k+1} = Y_k + V_k$$

$$V_{k+1} = \begin{cases} -2.5, & z_k = 1 \\ V_k + g, & z_k = 0. \end{cases}$$

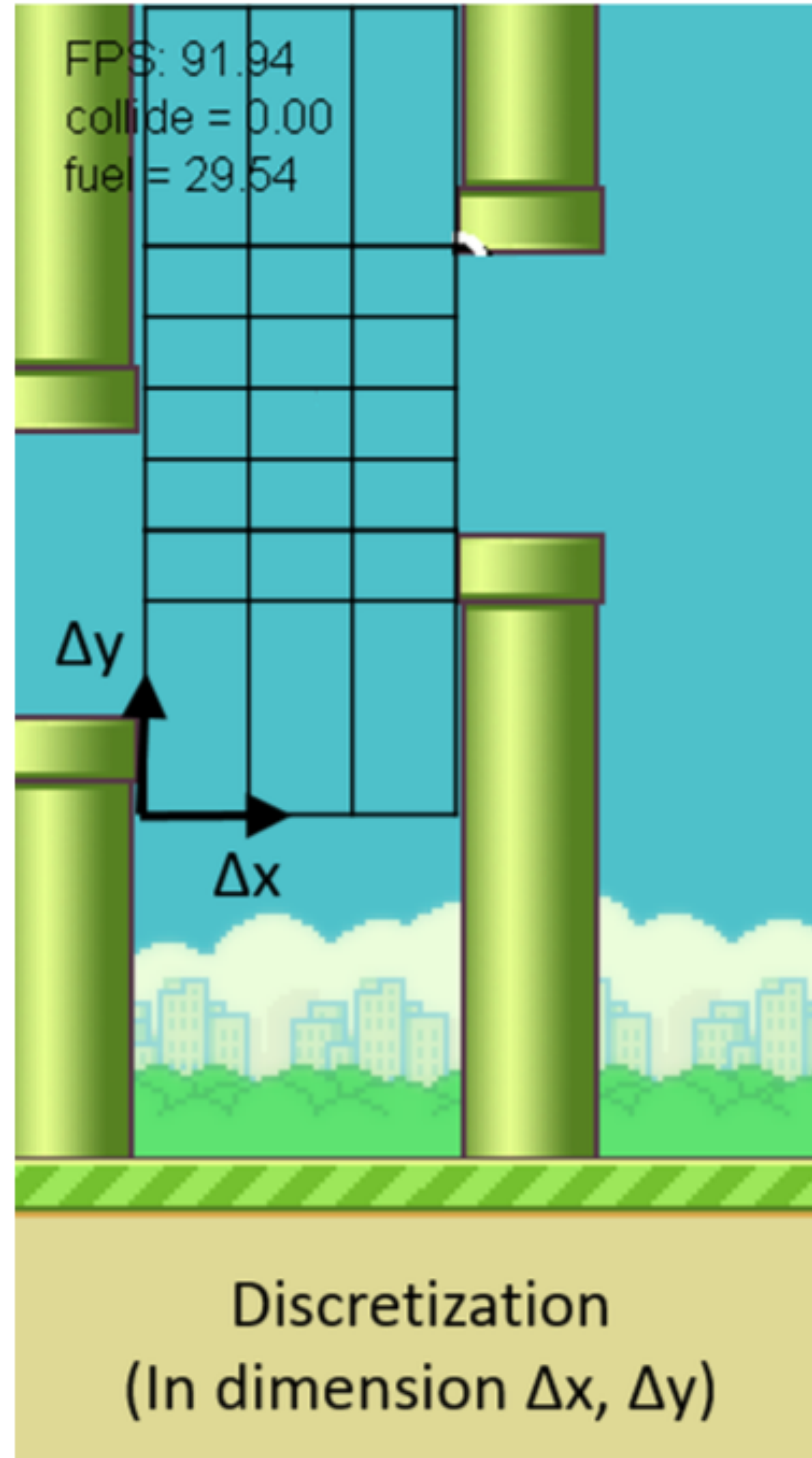
Y - vertical height (up -)

V - vertical velocity

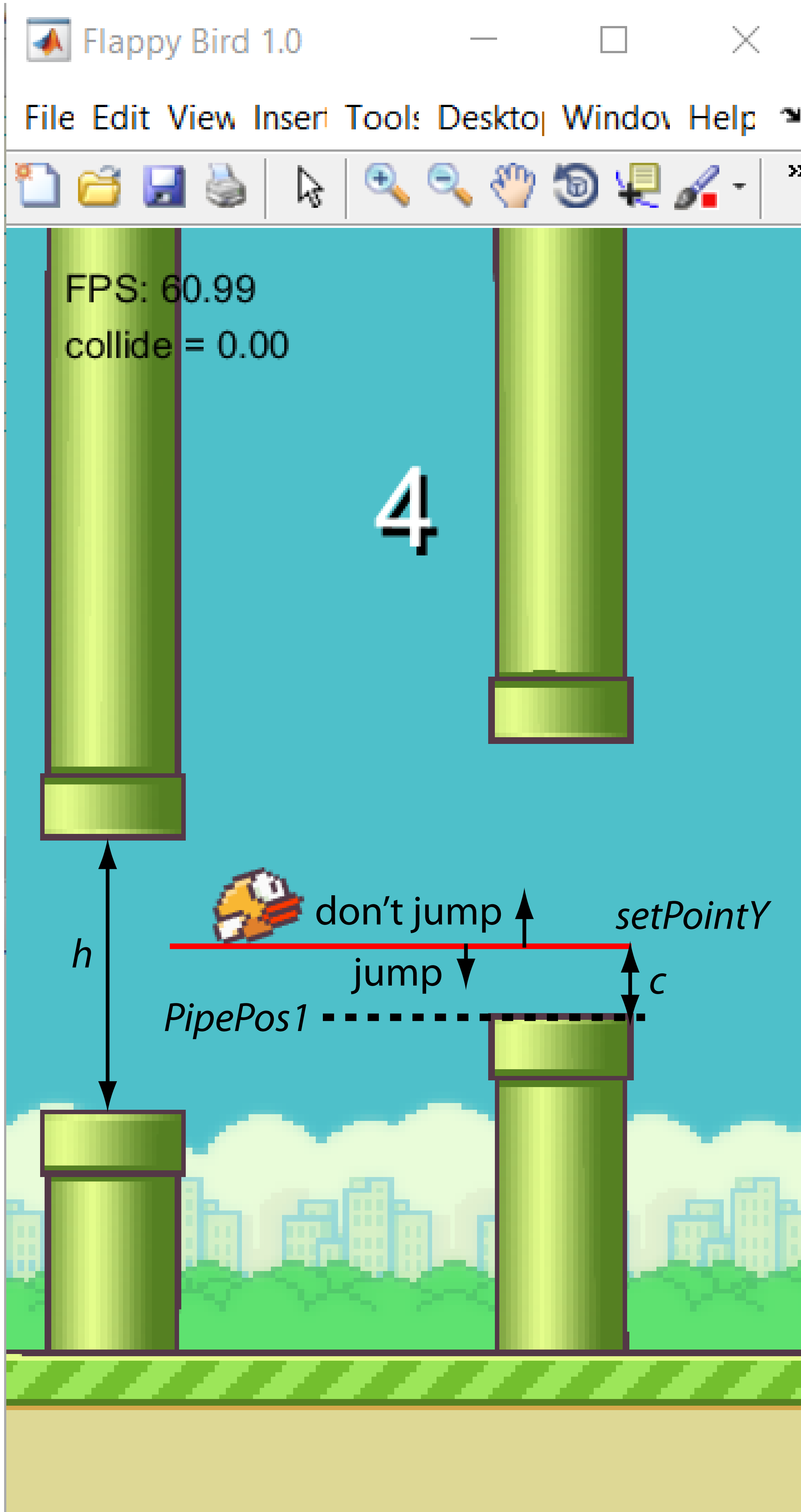
g - gravity (=0.1356)

z - control (flap or not flap)

constant horizontal velocity



#1: Heuristic controller & manual tuning



Set-point based control

Set-point is tuned.

Results: Heuristic controller & manual tuning

Run no.	Score
1	23
2	135
3	135
4	40
5	41
6	29
7	105
8	19
9	30
10	9

Average score 56.6/500

Results: Heuristic controller & manual tuning

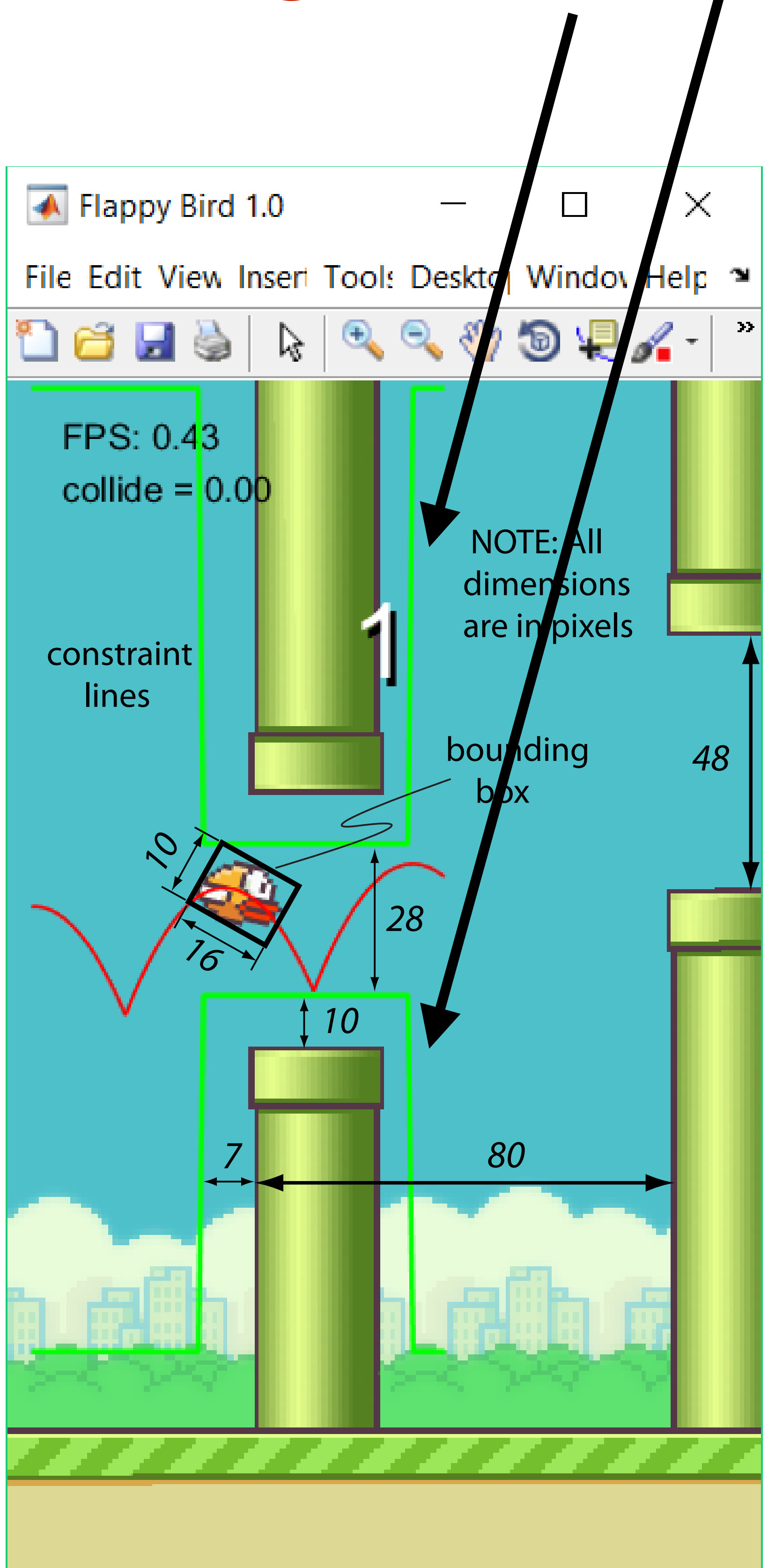
Controller 1:

Manually tuned controller

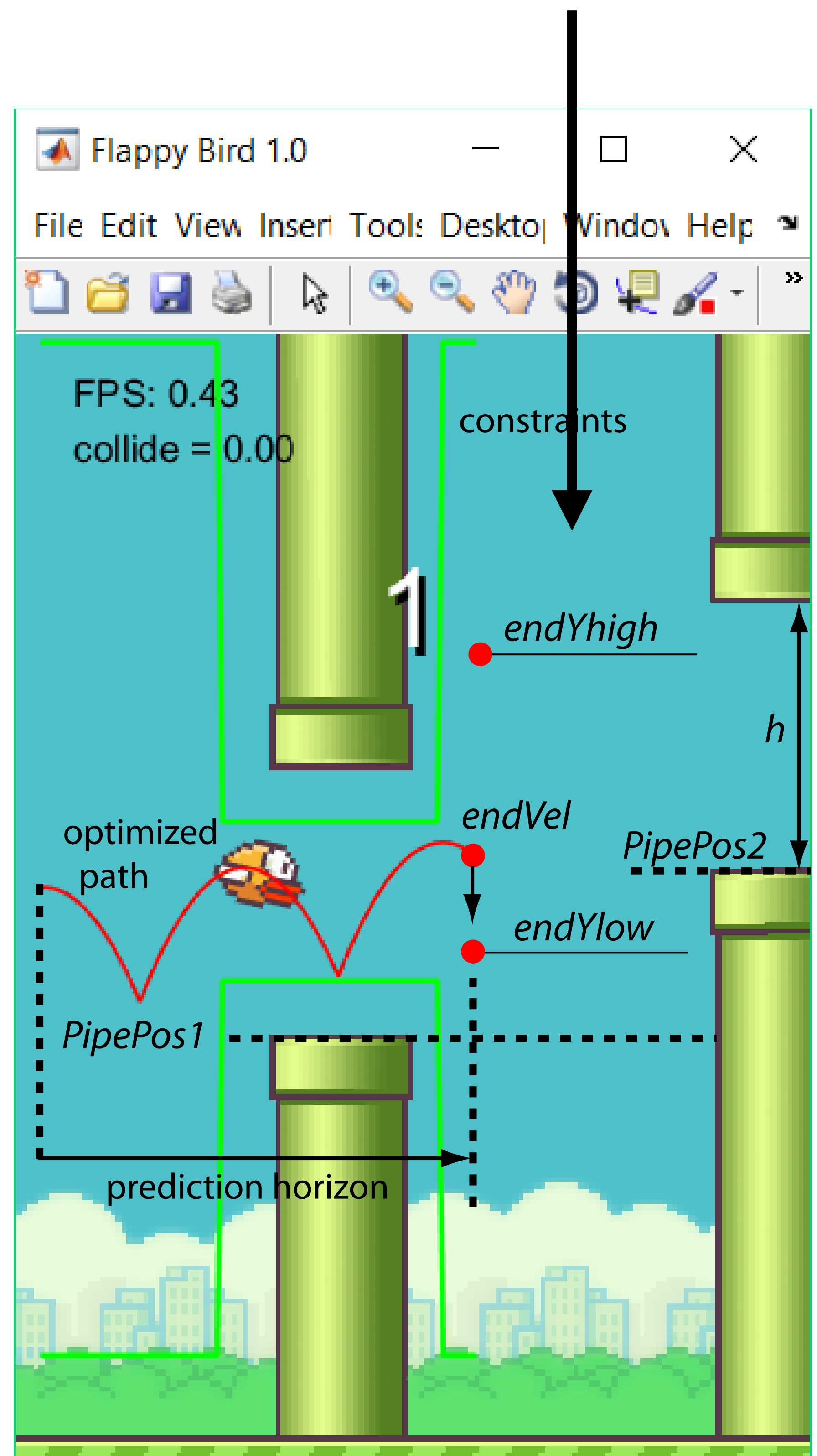
Highest score 23

#2: Optimization with manually tuned constraints

Bounding box constraints



Terminal constraints



#2: Optimization with manually tuned constraints

Minimize number of jumps

Input: Jump or not (z=0 or 1 resp.) for horizontal distance bet. pipes.

Constraints:

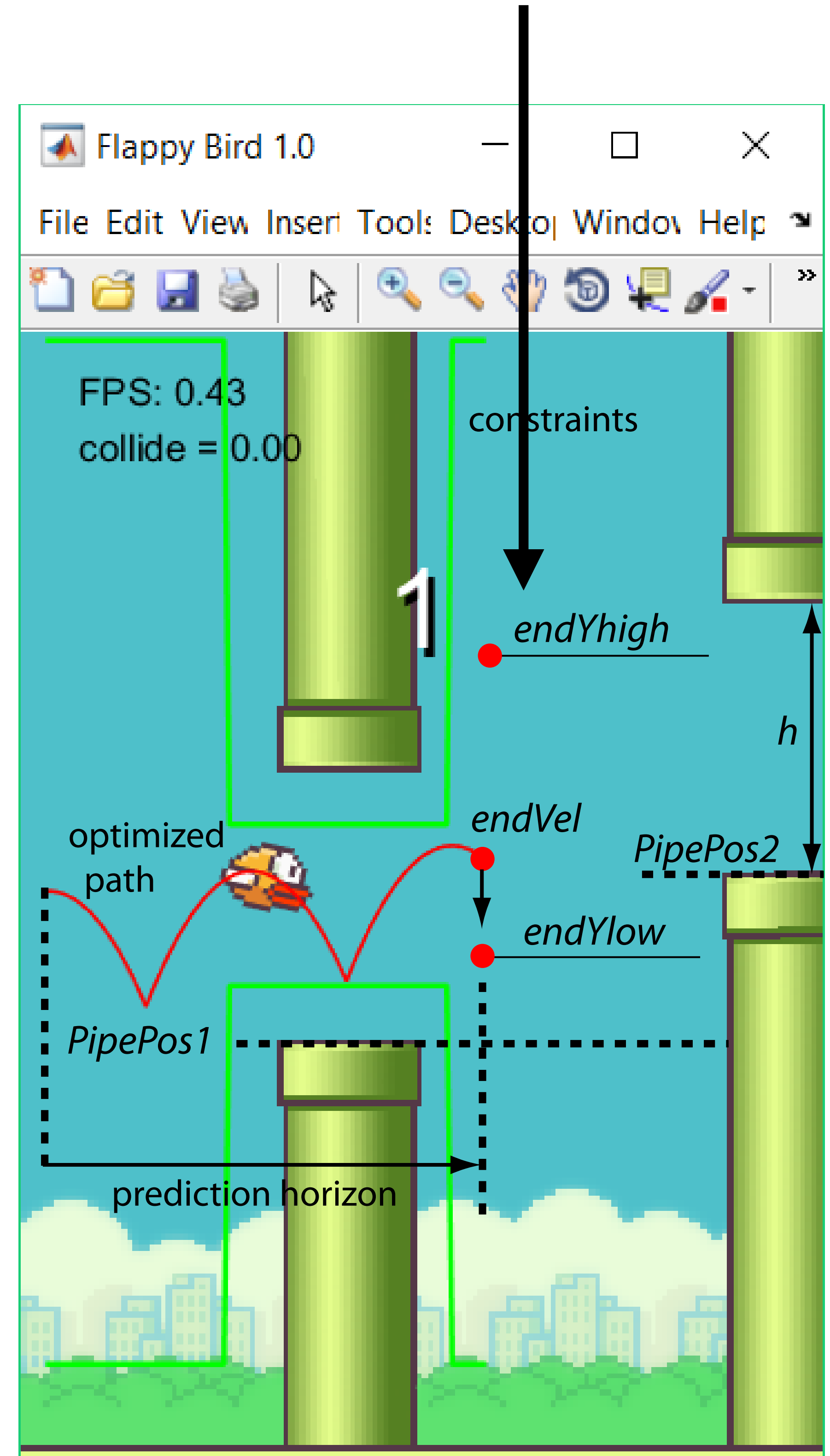
1) Physics (big M method)

2) Bounding box constraint (pipes)

3) Terminal constraints (exit)
[3 conditions parameters]

Mixed Integer Programming software
Gurobi (intlinprog)

Terminal constraints



Results: Optimization-based control, optimization horizon fixed

Run no.	Score	Avg. opt. time	Max. opt. time
1	500	0.1165	0.78766
2	500	0.1143	0.6863
3	500	0.1086	0.8101
4	500	0.1024	0.5053
5	500	0.1059	0.6243
6	500	0.1040	0.5273
7	500	0.1073	$\boxed{1.2596}$
8	500	0.1001	0.5508
9	500	0.0999	0.4946
10	500	0.1026	0.5173

Perfect score 500/500

Results: Optimization-based control, optimization horizon fixed

Controller 2:

Optimization-based controller with heuristic constraint tuning

Highest Score 6473

#3: Model Predictive Control

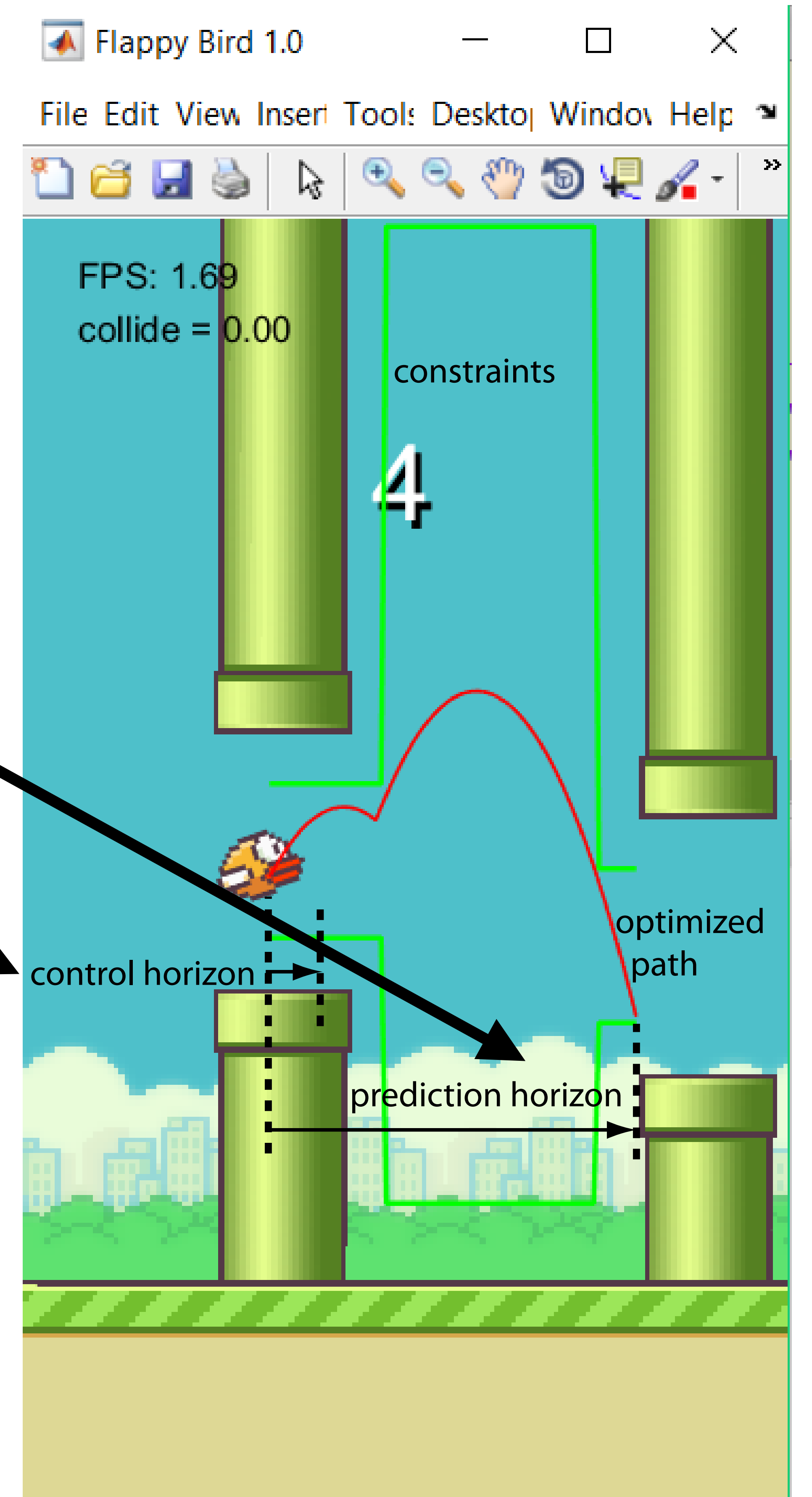
Same as #2 but with **TWO** differences:

1) no terminal constraint

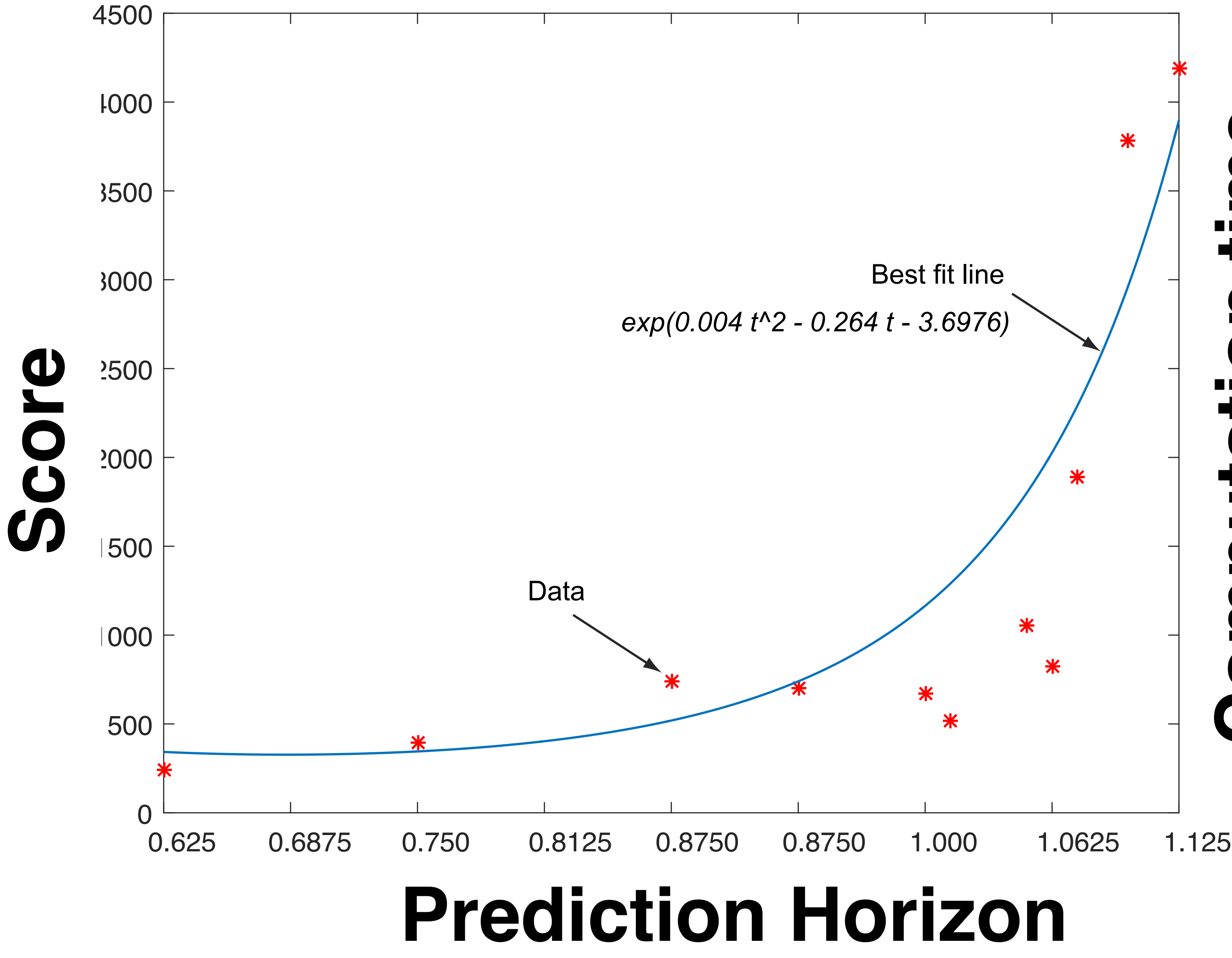
2) prediction horizon (n),

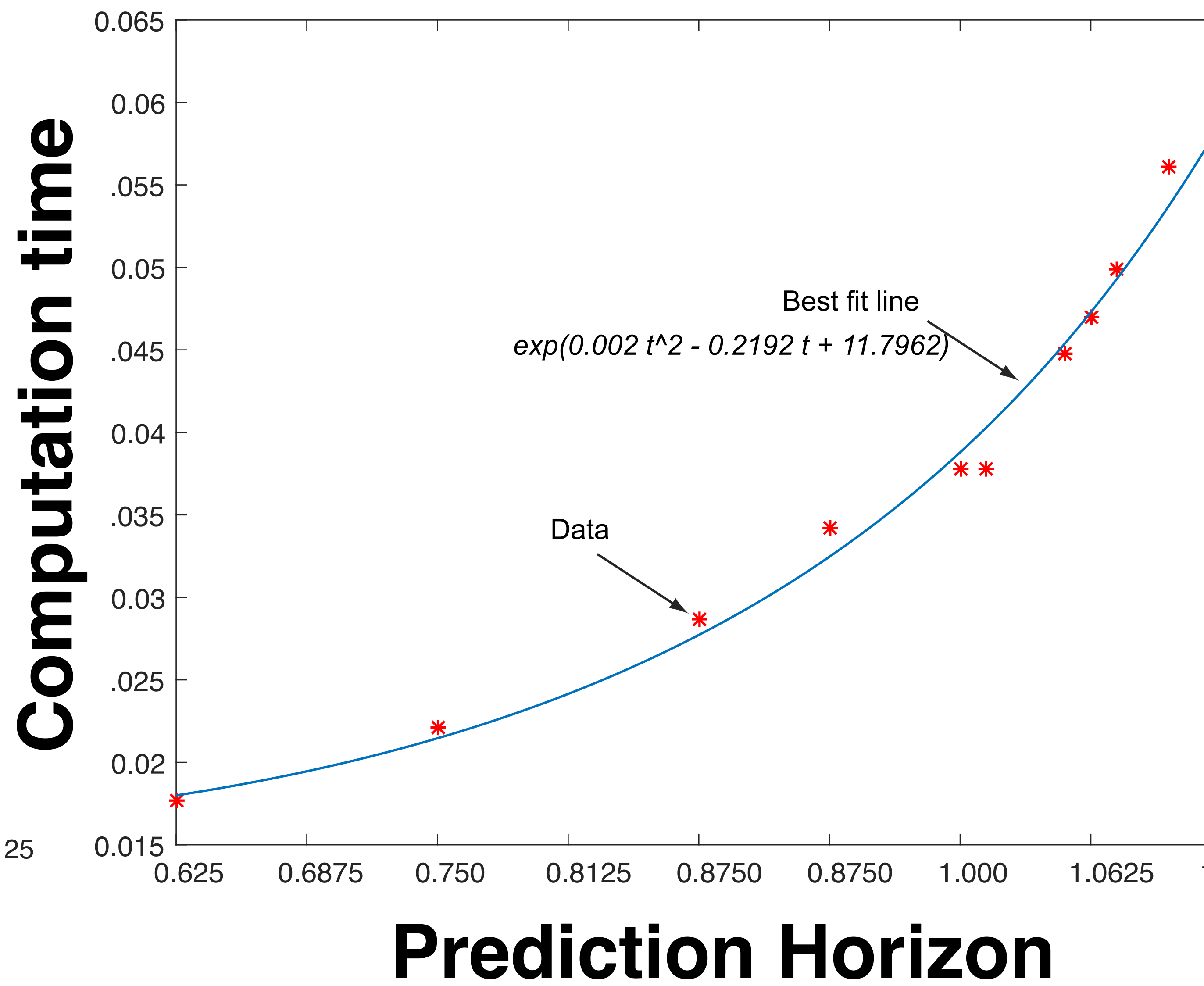
control horizon is 1 step.

[n is the only free parameter]



Results: Model Predictive Controller





Results: Model Predictive Controller (with optimum prediction horizon of 90)

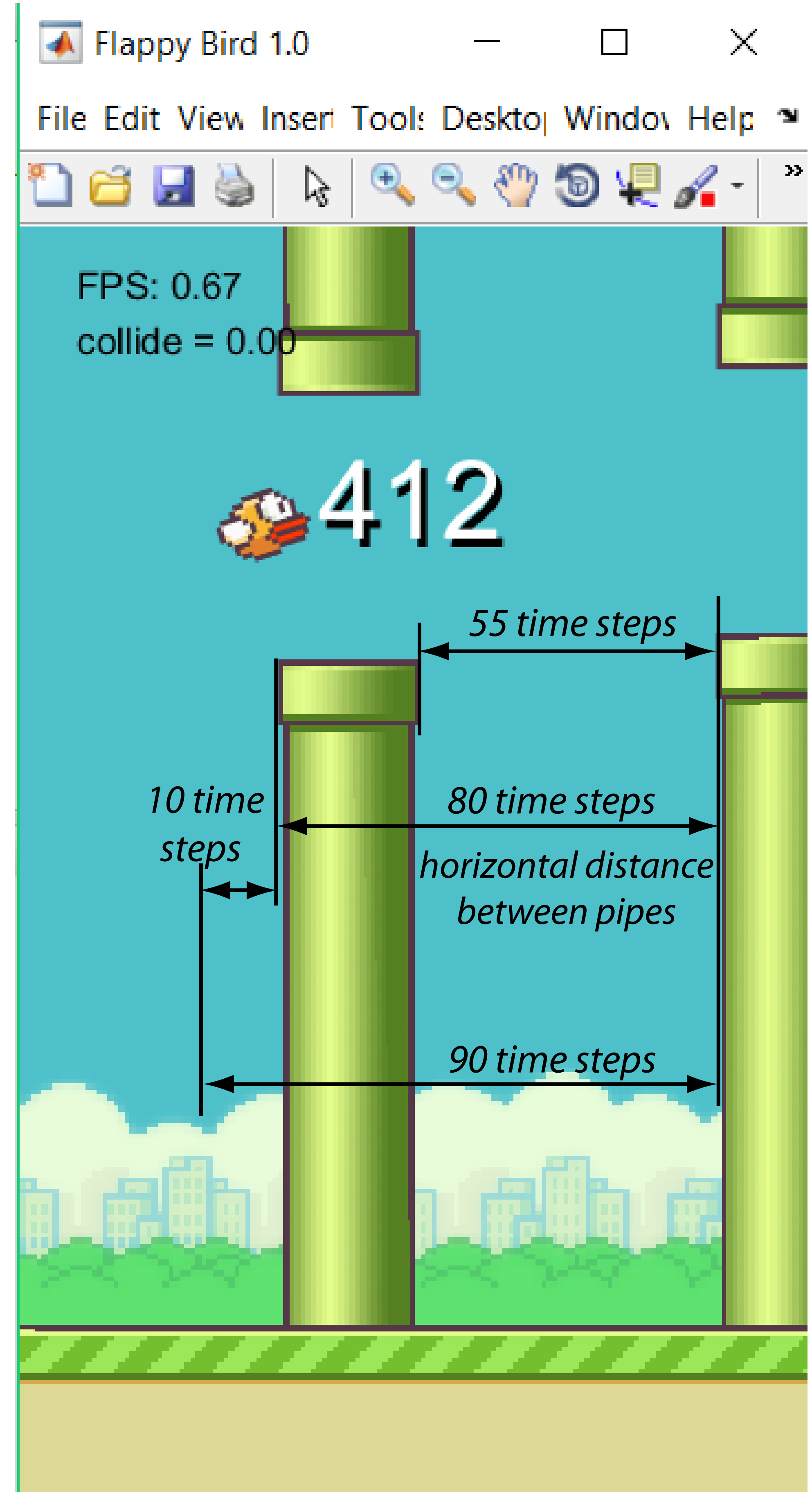
Run no.	Score	Avg. opt. time	Max. opt time
1	500	0.0671	3.591
2	500	0.0641	3.059
3	500	0.0640	2.8842
4	500	0.0596	2.9635
5	80	0.0550	0.9668
6	500	0.0613	2.9596
7	500	0.0573	3.0536
8	500	0.0680	3.8055
9	500	0.0567	1.9173
10	106	0.0616	1.7036

Average score 419/500

Results: Model Predictive Controller

Optimal prediction window is 90
~1.125 horizontal distance between pipes

Key message: Need to plan slightly beyond the next pipe



Results: Model Predictive Controller

Controller 3:

Model Predictive Controller

Highest Score 3961

Discussion

	Heuristic controller	Optimization	MPC
Score (10 runs, max 500 pipes)	56.6	500	419
Worst case time (sec)	~0	1.3	3.9
Tuning	Trial and error tuing	Trial and error tuning	Can be automated

Conclusion

- Position and speed on exiting the pipe seems to be key factors for good performance
- Optimization/MPC are too slow for real time implementation
- MPC best compromise between scores and need for intuitive tuning