# Computing the Physical Parameters of Rigid-body Motion from Video

Kiran S. Bhat, Steven M. Seitz, Jovan Popovic, and Pradeep K. Khosla

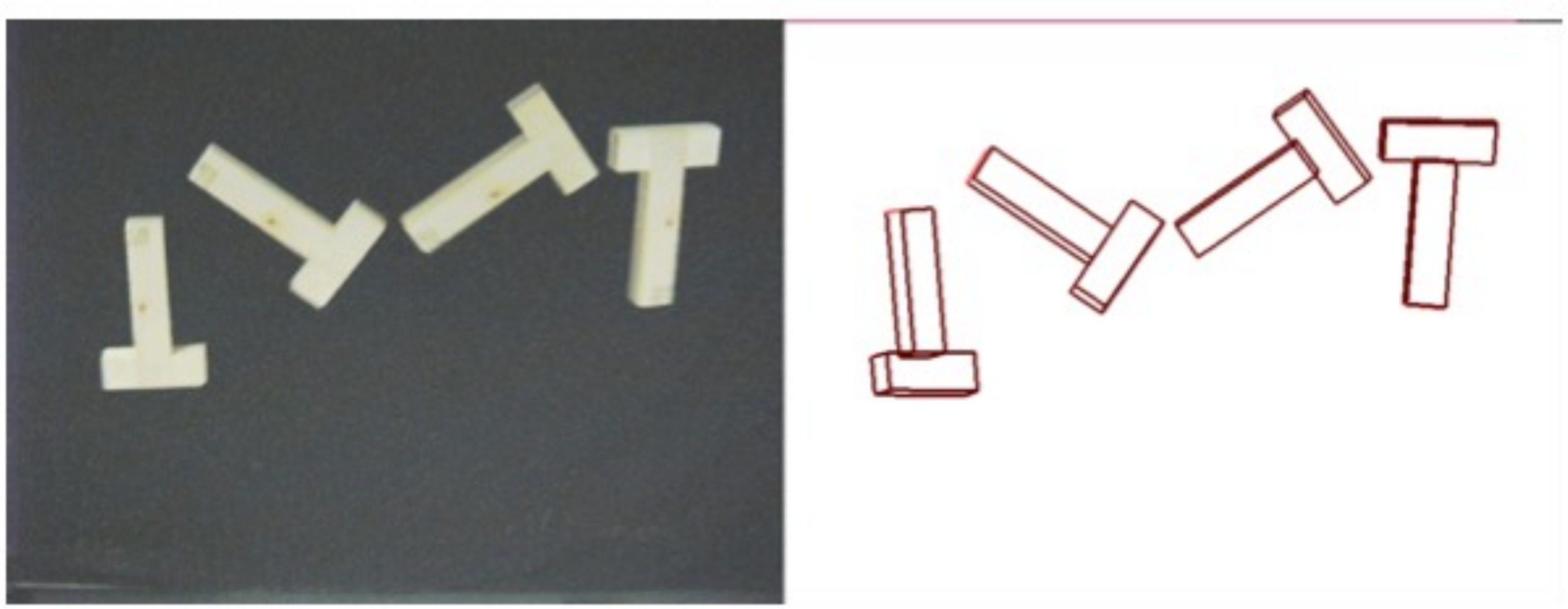
Presenter: Jinyan Guan

# Overview

- This paper presents an **optimization framework** for estimating the **motion** and underlying **physical parameters** of a **rigid body** in **free flight** from video.

# Problem Statement

- Goal: infer the physical parameters underlying the motion of the rigid body in free flight from a pre-recorded video sequence of its motion.



# Simulating Motion of Rigid Body

- State $\mathrm{q}(t)$ of tumbling motion of a rigid body in free flight:
  - Position: $\mathbf{X}(t)$
  - Orientation: $\boldsymbol{\theta}(t)$ (in quaternion form)
  - Linear velocity: $\mathbf{V}(t)$
  - Angular velocity: $\boldsymbol{\omega}(t)$
- Simulation Function: $\mathbf{F}(t, \mathbf{q}(t))$

$$\boldsymbol{F}(t, \boldsymbol{q}(t)) = \frac{d}{dt}\left(\boldsymbol{q}(t)\right) = \frac{d}{dt}\begin{pmatrix} \boldsymbol{X}(t) \\ \boldsymbol{\theta}(t) \\ \boldsymbol{V}(t) \\ \boldsymbol{\omega}(t) \end{pmatrix} = \begin{bmatrix} \boldsymbol{V}(t) \\ \frac{1}{2}(\boldsymbol{\theta}(t) * \boldsymbol{\omega}(t)) \\ \boldsymbol{g} \\ -\boldsymbol{I}(t)^{-1}\left(\frac{d\boldsymbol{I}(t)}{dt}\boldsymbol{\omega}(t)\right) \end{bmatrix} \qquad (4.1)$$

# Approach

- Assumptions:
  - Object shape and mass distribution are known (inertia matrix is known)
- Model:
  - Tumbling dynamics using ordinary differential equations (ODE)
  - State q(t): $\boldsymbol{q}(t_f) = \boldsymbol{q}(t_0) + \int_{t_0}^{t_f} \boldsymbol{F}(t, \boldsymbol{q}(t))dt$
- Optimization:
  - Seeks to match the resulting motion with the frames of the video sequence.
  - Solve for $\mathbf{q}(t_0)$

# Parameters

- Object:
  - Intrinsic: object shape, mass distribution and center of mass
  - Extrinsic: initial position, orientation, velocity and angular velocity
- Environment: gravity direction and air drag
- Camera:
  - Intrinsic: focal length, principle point, etc.
  - Extrinsic: position and orientation

# Interested Parameters

- Object:
  - Intrinsic: object shape, mass distribution and center of mass
  - Extrinsic: initial position, orientation, velocity and angular velocity
- Environment: gravity direction and air drag
- Camera:
  - Intrinsic: focal length, principle point, etc.
  - Extrinsic: position and orientation

# Estimating Parameters from Video

- Simulation parameters: $\mathbf{p} = (\mathbf{p}_{obj}, \mathbf{p}_{env}) \longrightarrow \mathbf{q}(t_0)$
- Solve for $\mathbf{p}$ by minimizing the least square error between silhouettes from video and silhouettes from the simulation on each frame.
- $(\mathbf{X}(t_0), \theta(t_0)), \ldots, (\mathbf{X}(t_k), \theta(t_k))$is the sequence of poses computed from a sequence of k frames.
- Objective Function for 3D data:

$$\min_{\boldsymbol{\omega}(t_0), \boldsymbol{v}(t_0)} \sum_{i=t_0, t_1 \ldots t_k} (\boldsymbol{X}(i) - \boldsymbol{X}^s(i))^2 + (\boldsymbol{\theta}(i) - \boldsymbol{\theta}^s(i))^2 \qquad (4.3)$$

# Optimization

- Object function: shape-based metrics to compare the real and simulated motions
- Works offline and takes account of all frames
- For a given set of parameters, the optimizer simulates the motion to compute the bounding boxes and silhouette at each frame.
- They provide an initial estimate of the parameters and use a gradient descent to update it

# Optimization

- Objective Function:

$$E = \min_{\mathbf{p}} \sum_{i=t_1,t_2\ldots t_f} \left(\mathbf{S}^v(i) - \mathbf{S}^s(i,\mathbf{p})\right)^2 \tag{5.1}$$

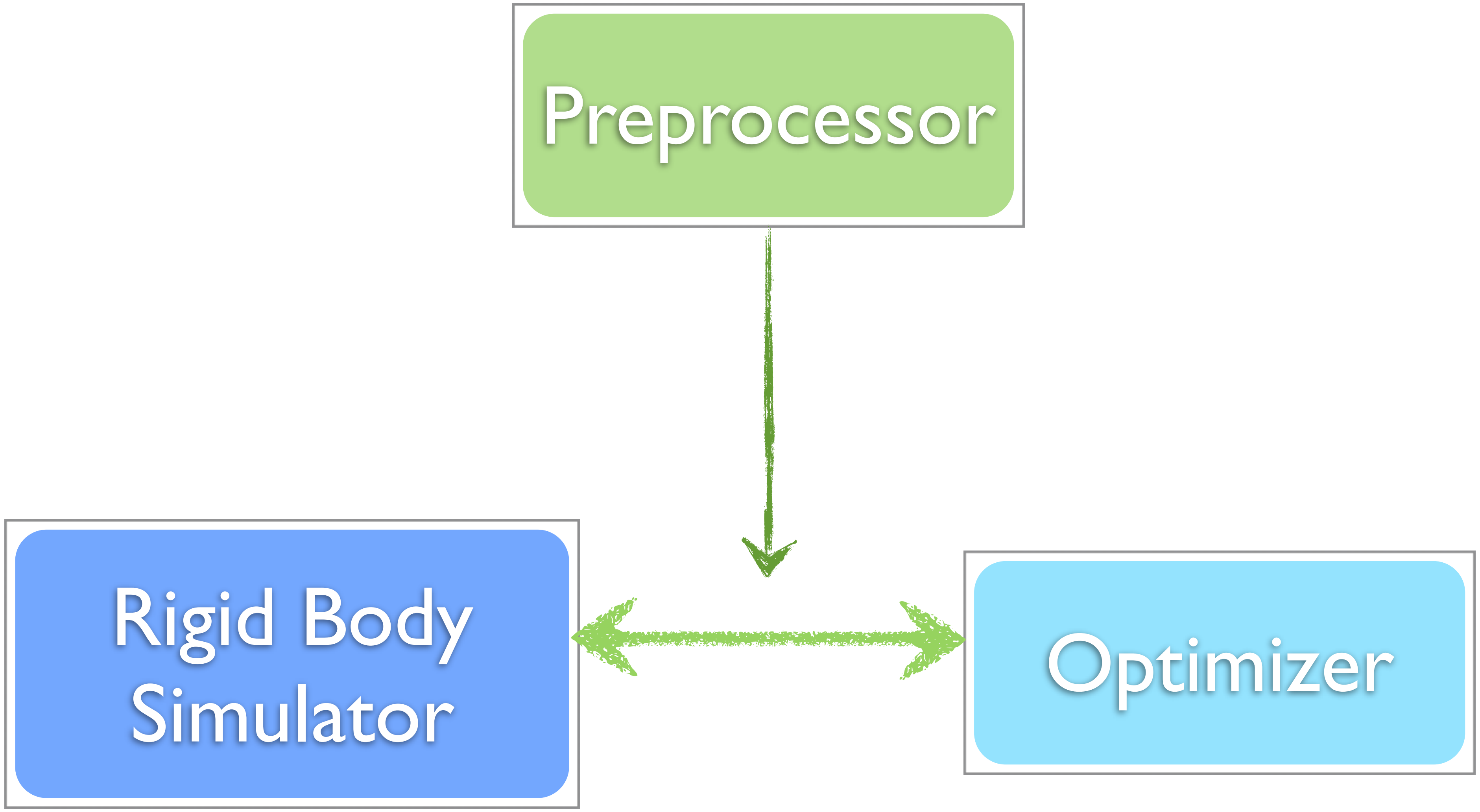
- Update rule:

$$\mathbf{p} = \mathbf{p} + \lambda \frac{\partial E}{\partial \mathbf{p}} \tag{5.2}$$

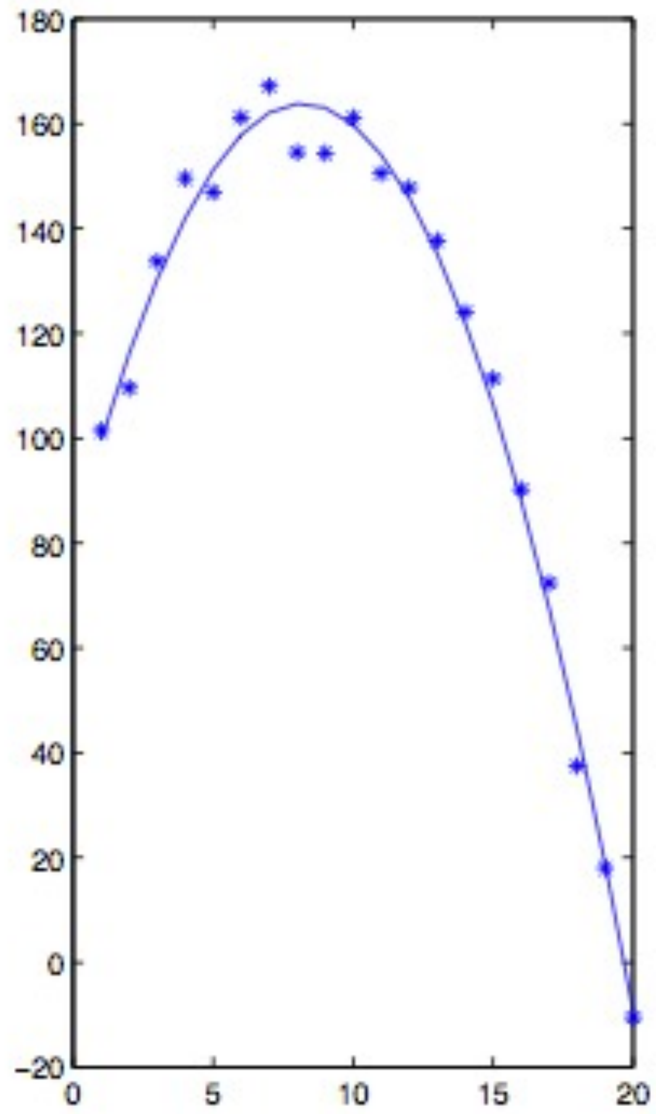
# Gradient Computation

- Use finite different to compute: $\frac{\partial \mathbf{S}}{\partial \mathbf{p}}$
    - It's very slow
    - Hard to determine a robust step size
- A hybrid approach:
    - First, analytically compute: $\frac{\partial \mathbf{q}(t)}{\partial \mathbf{p}}$
    - Second, use finite difference compute $\frac{\partial \mathbf{S}}{\partial \mathbf{q}}$
    - Use chain rule: $\frac{\partial \mathbf{S}}{\partial \mathbf{p}} = \frac{\partial \mathbf{S}}{\partial \mathbf{q}} \frac{\partial \mathbf{q}(t)}{\partial \mathbf{p}}$

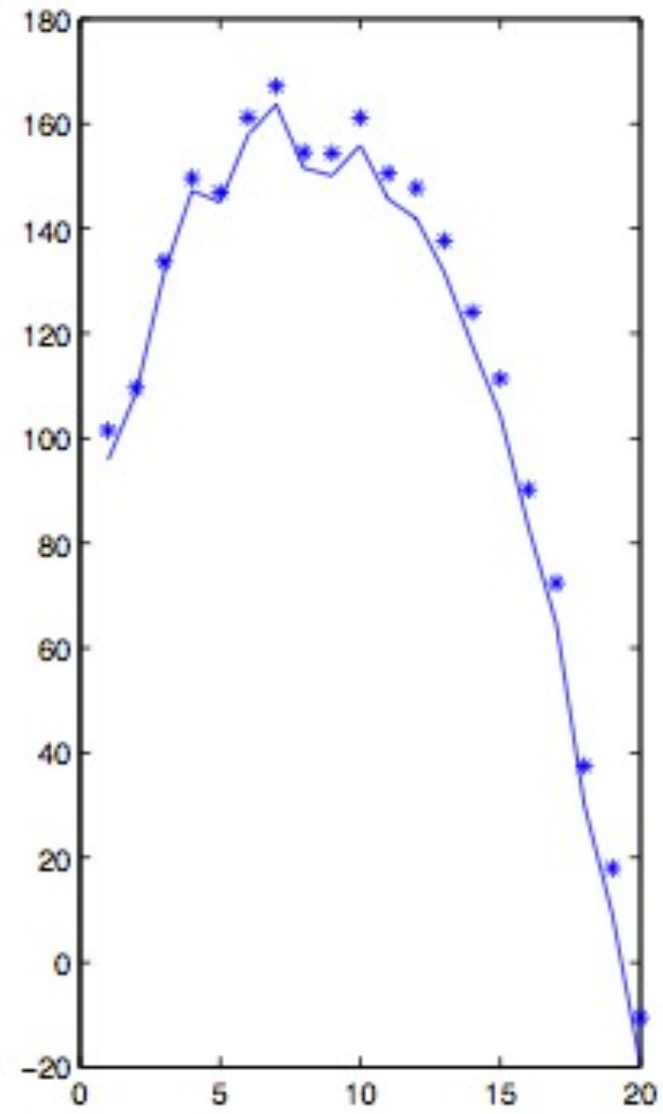
# Overall Architecture



# Results

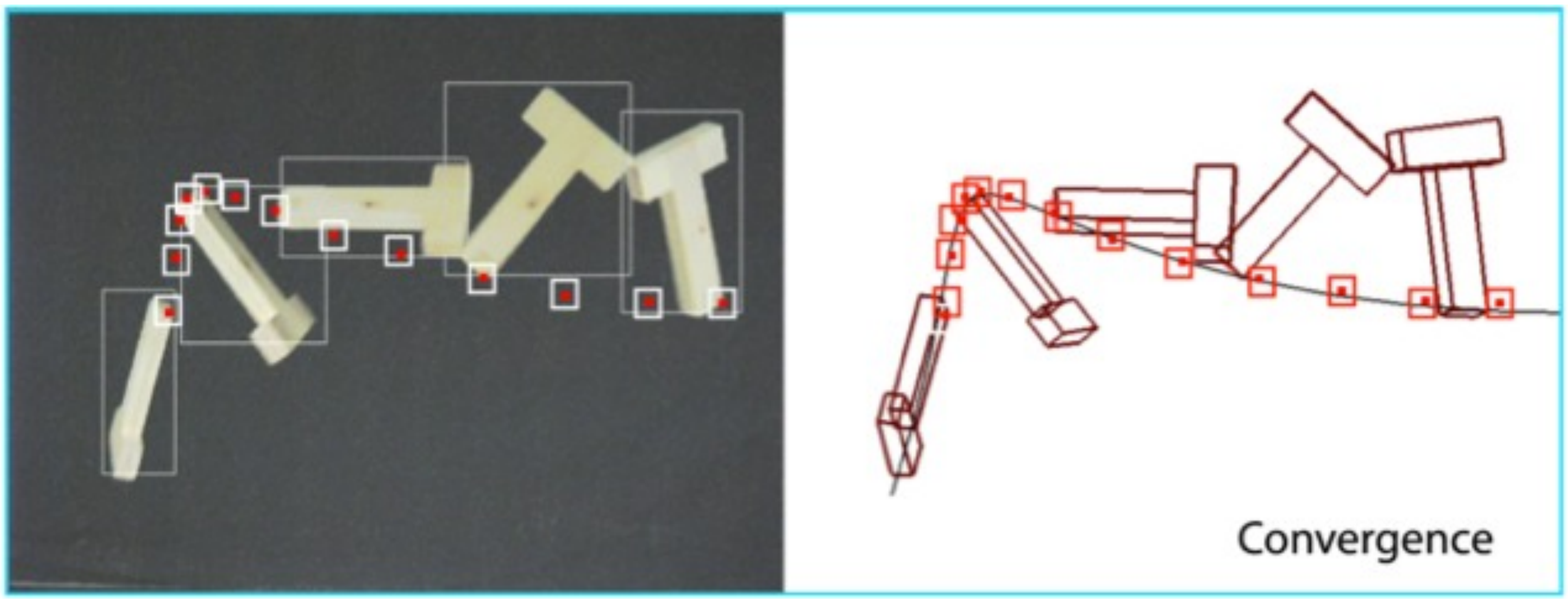



Optimization



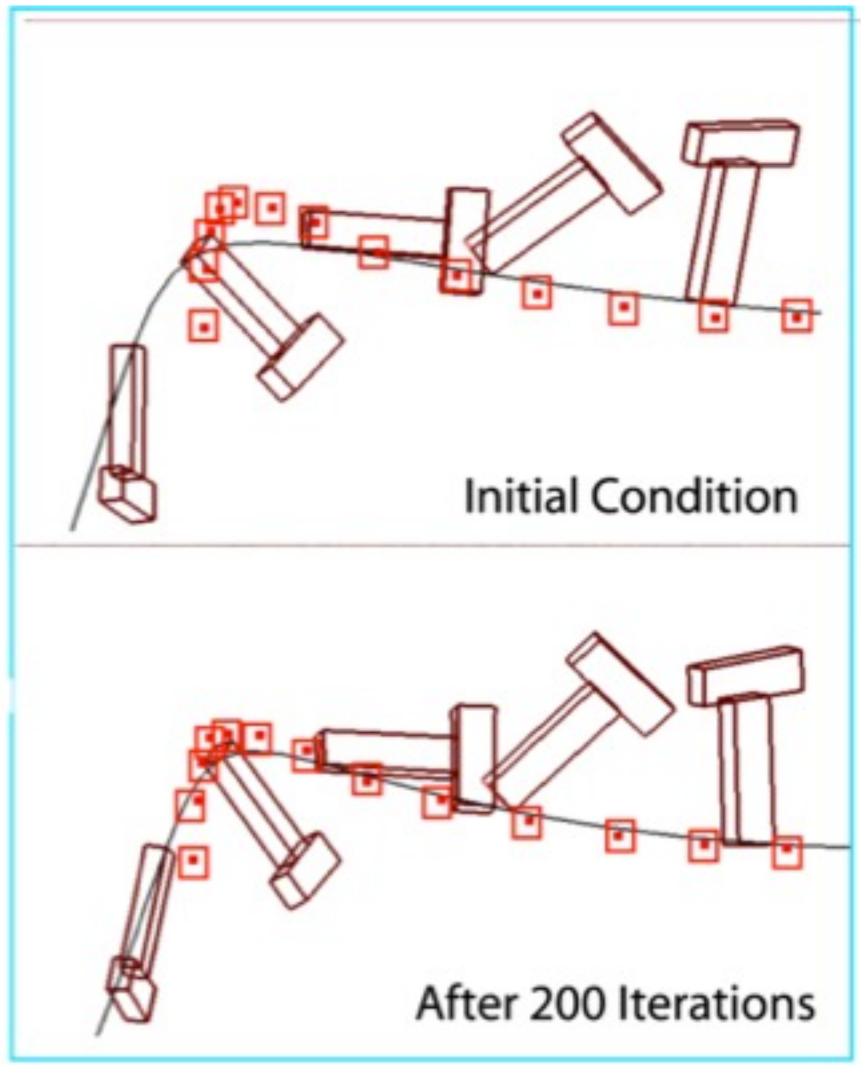

Kalman filter

# Results



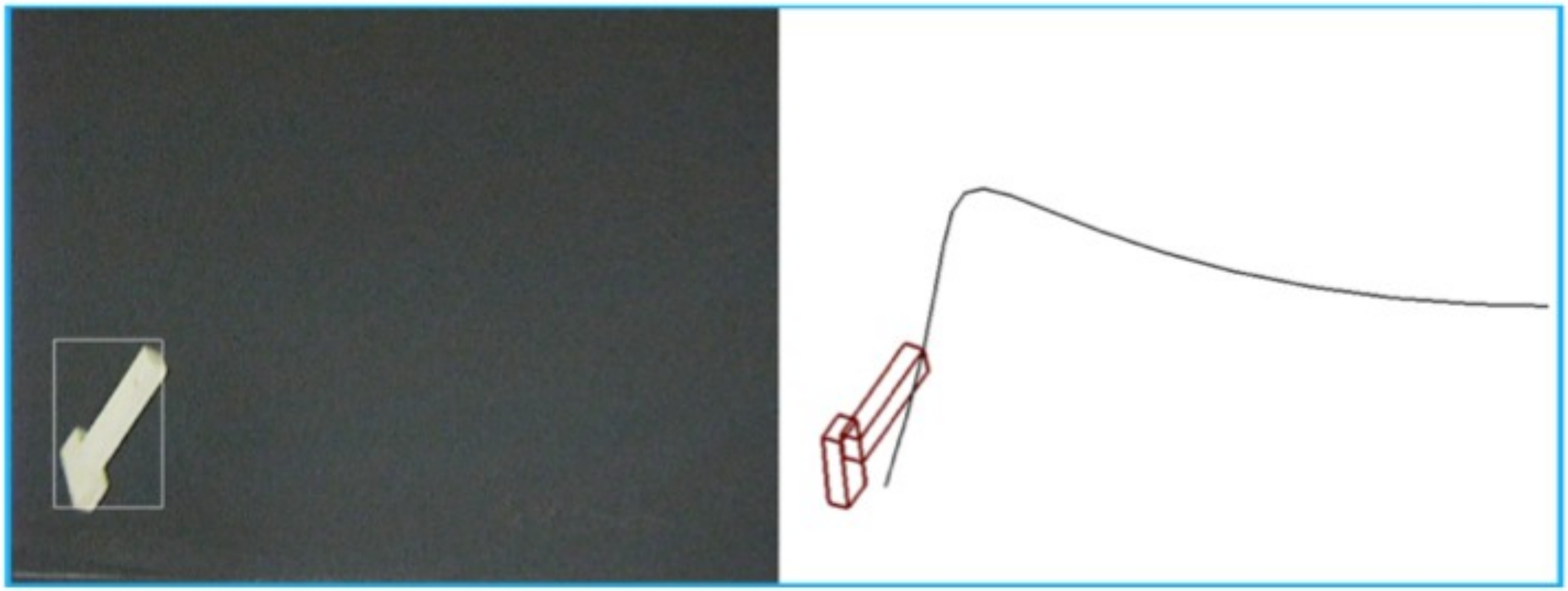

Frames from video and the corresponding physical simulation generated

# Results



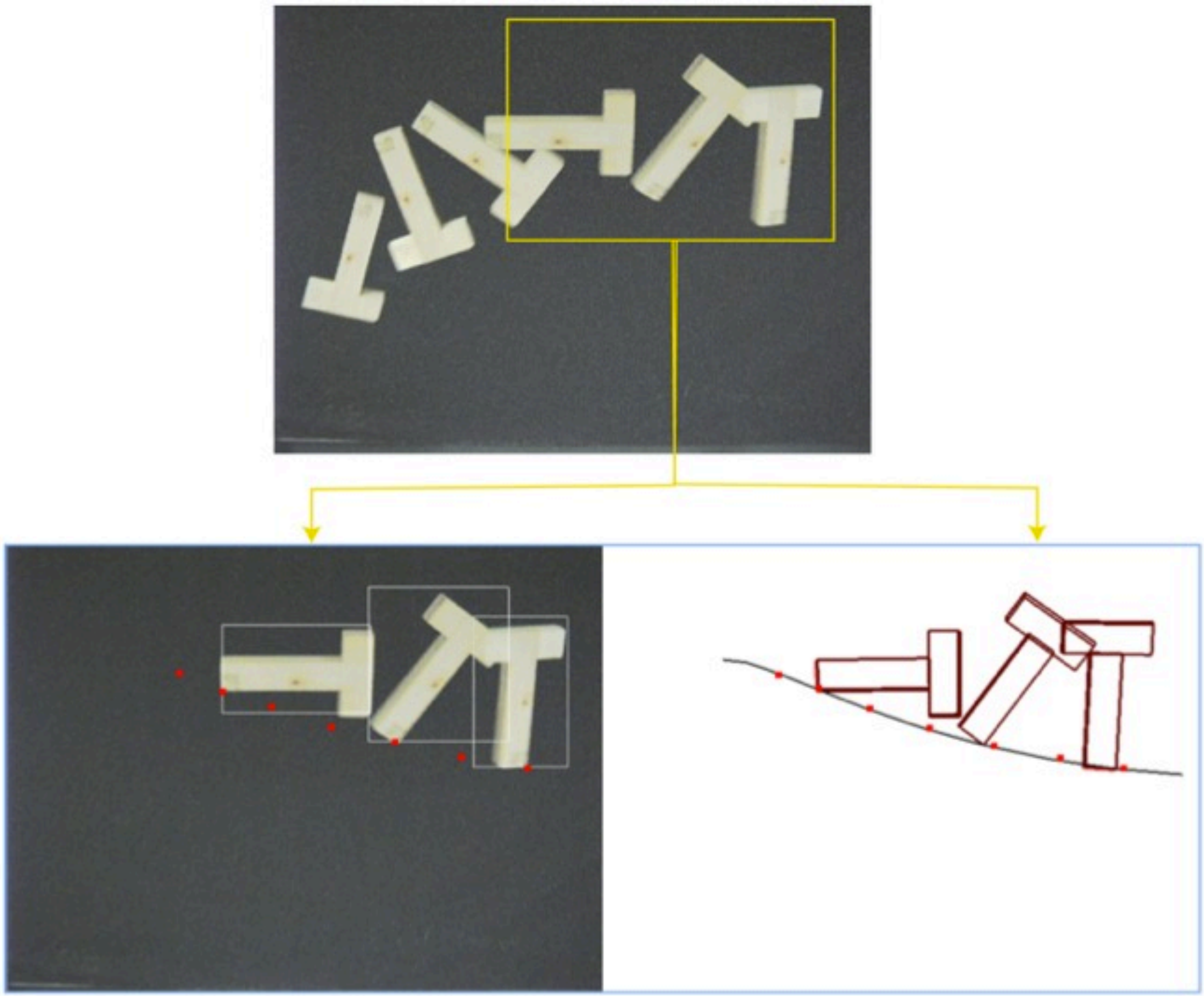

Different stages of optimization

# Results

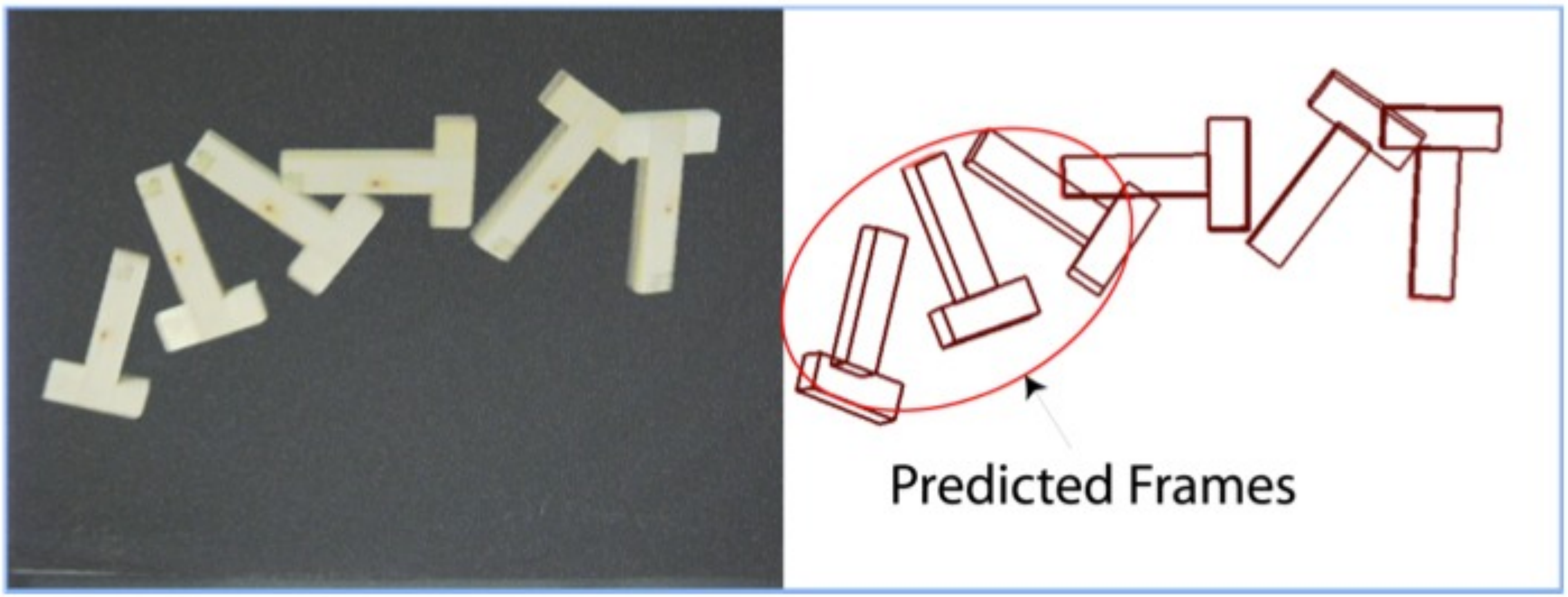


Limitation of silhouette based metric

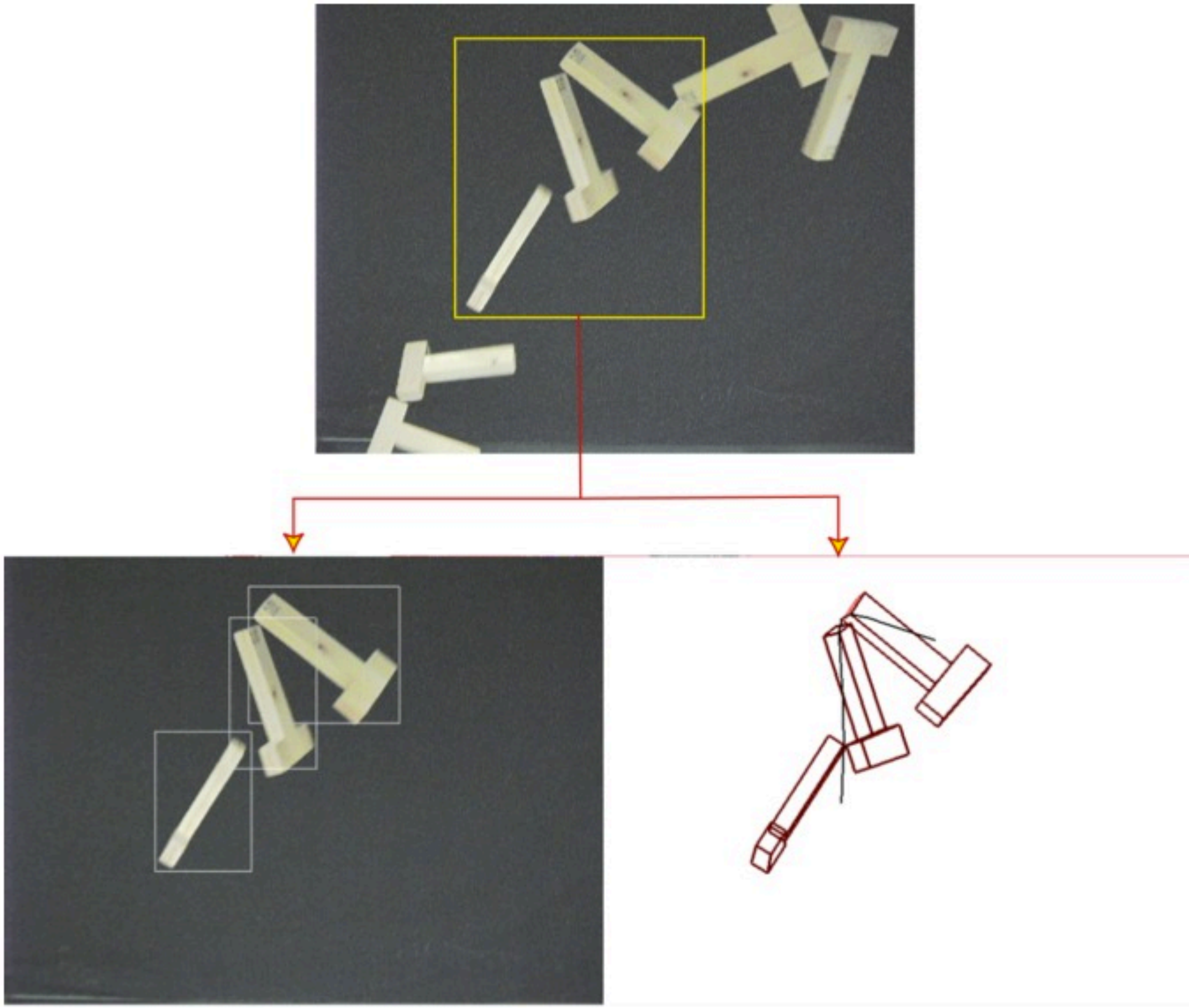
# Results



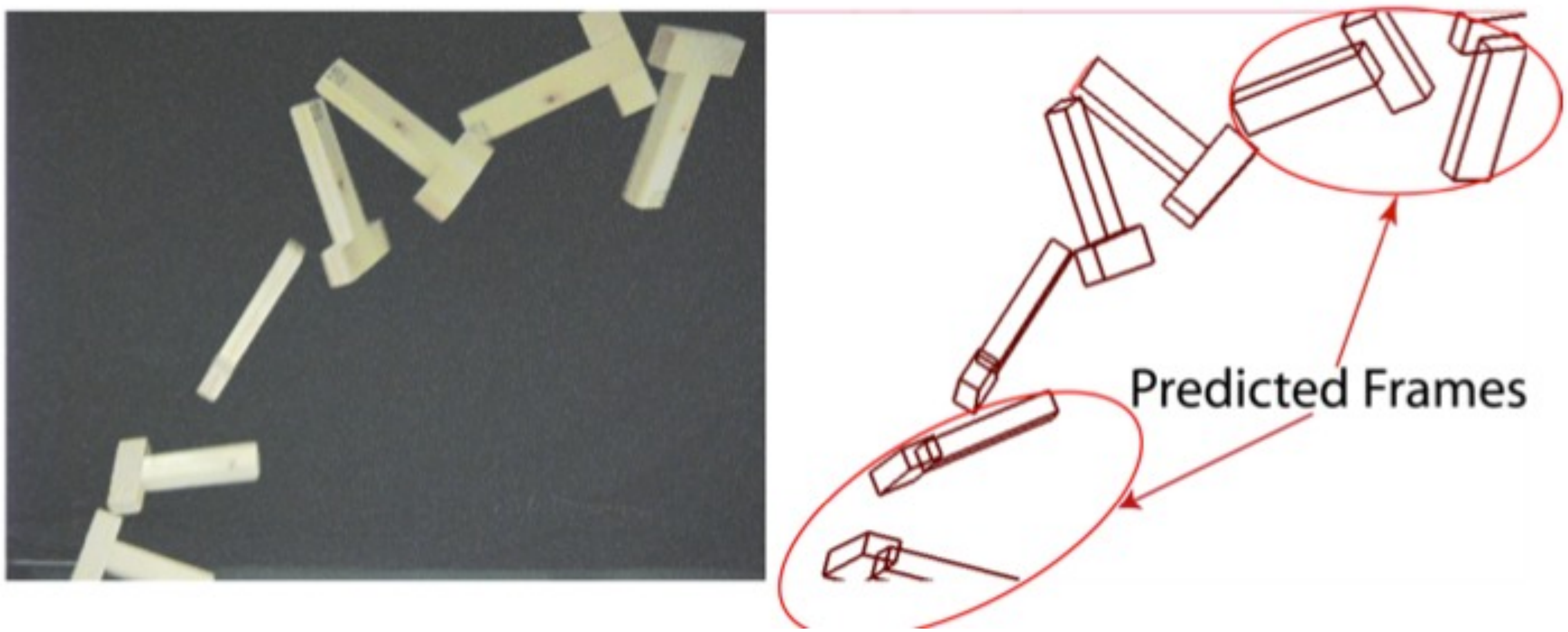
# Results



# Results

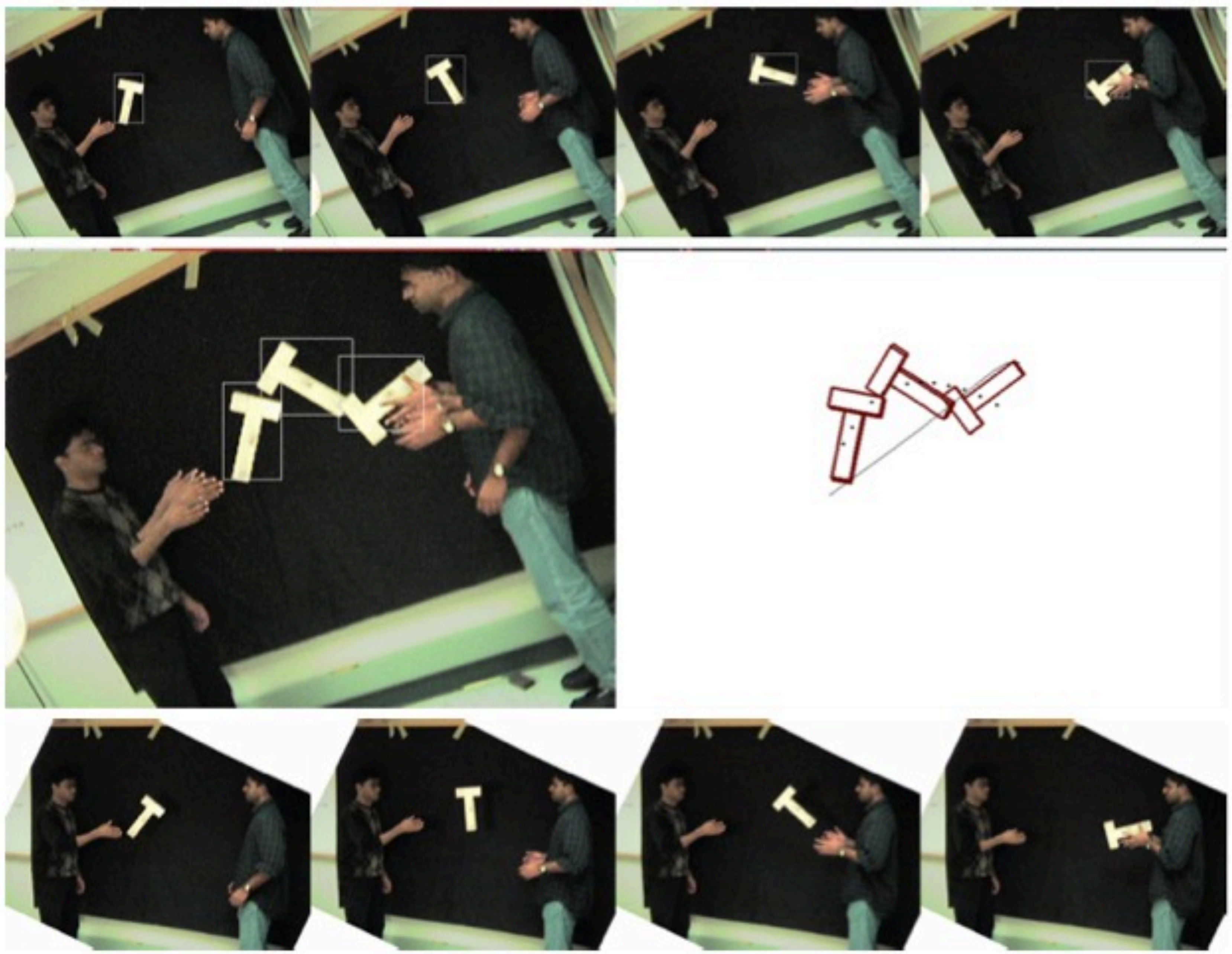


# Results




Small error propagates to larger errors in the predicted frames

# Results



Correction for camera roll

# Benefits of Estimating Physical parameters

- An accurate physical model actually **simplifies** the task of recovering kinematics.
- Enables the behavior of the object to be **predicted** in new or unseen condition.
- Allows to predict how the object would behave in different conditions

# Questions?

# Procedure

- Step 1: Preprocess the video to obtain a background model of the scene.
- Step 2: Segment the moving object and compute the bounding box **B** and silhouette **S** at each frame.
- Step 3: Optimization