# Meshless Deformations Based on Shape Matching

## by Matthias Müller, Bruno Heidelberger, Matthias Teschner, Markus Gross

Presentation autor: Andrej Čičmanský

# Content

# Abstract

- new approach for simulating deformable objects
- handles point objects and does not need connectivity information
- does not require any pre-processing
- unconditional stability of the dynamic simulation make the approach particularly interesting for games

# Introduction:
# What we need

- Efficiency
- Stability
- Controllability

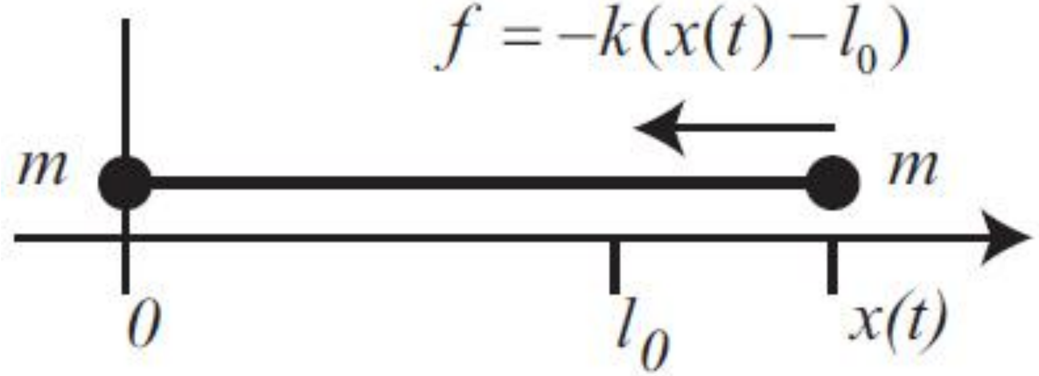
# Introduction: Contributions

- pulling a deformed geometry towards a well-defined goal
- degree of details are varied using linear and quadratic deformation modes
- large variety of objects can be handled
- stable under all circumstances and for all deformed geometry configurations

# Meshless Animation

- Newton's second law of motion is basis for many physically-based simulation techniques  $F = ma$
- to compute object locations, the accelerations and velocities are numerically integrated over time
- implicit integration – stability / computationally expensive
- explicit integration - faster to compute / not so stable

# Meshless Animation: Explicit Numerical Integration

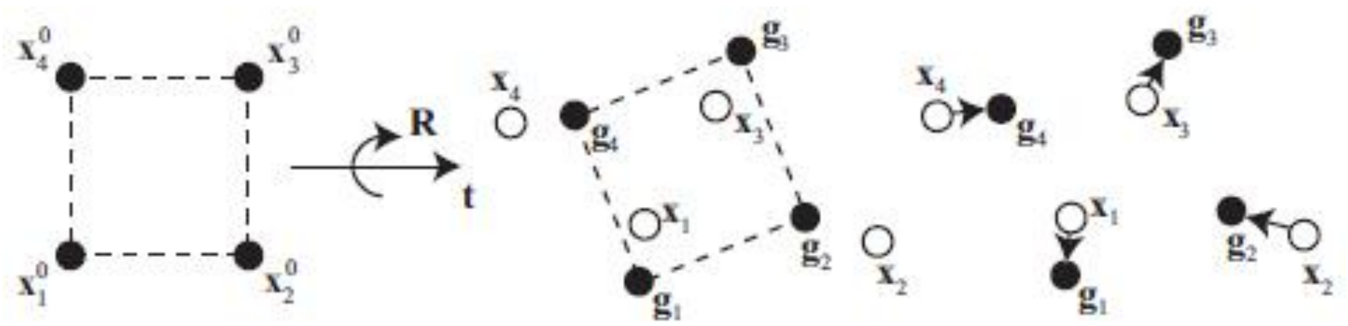



$$\begin{aligned} v(t+h) &= v(t) + h\frac{-k(x(t)-l_0)}{m} \\ x(t+h) &= x(t) + hv(t+h), \end{aligned}$$

$$A = \begin{bmatrix} 1 & -\frac{kh}{m} \\ h & 1 - \frac{h^2k}{m} \end{bmatrix}$$

# Meshless Animation: The Algorithm

- only need set of particles with masses $m_i$ and an initial configuration $x_i$
- without particle-particle interactions
- each time step, each particle is pulled towards its goal position $g_i$



# Meshless Animation: Shape Matching 1

- two sets of points $x_i^0$ and $x_i$
- find the rotation matrix R and the translation vectors t and $t_0$ which minimize

$$\sum_i w_i(\mathbf{R}(\mathbf{x}_i^0 - \mathbf{t}_0) + \mathbf{t} - \mathbf{x}_i)^2$$

- $w_i$ are weights of individual points

# Meshless Animation: Shape Matching 2

$$\mathbf{A} = \left(\sum_i m_i \mathbf{p}_i \mathbf{q}_i^T\right)\left(\sum_i m_i \mathbf{q}_i \mathbf{q}_i^T\right)^{-1} = \mathbf{A}_{pq}\mathbf{A}_{qq}.$$

- $A_{pq}$ = RS
- S - symmetric part
- R - rotational part

- Goal position:

$$\mathbf{g}_i = \mathbf{R}(\mathbf{x}_i^0 - \mathbf{x}_{\mathrm{cm}}^0) + \mathbf{x}_{\mathrm{cm}}$$

# Meshless Animation: Integration

$$\mathbf{v}_i(t+h) = \mathbf{v}_i(t) + \alpha \frac{\mathbf{g}_i(t) - \mathbf{x}_i(t)}{h} + h f_{\text{ext}}(t)/m_i$$

$$\mathbf{x}_i(t+h) = \mathbf{x}_i(t) + h\mathbf{v}_i(t+h)$$

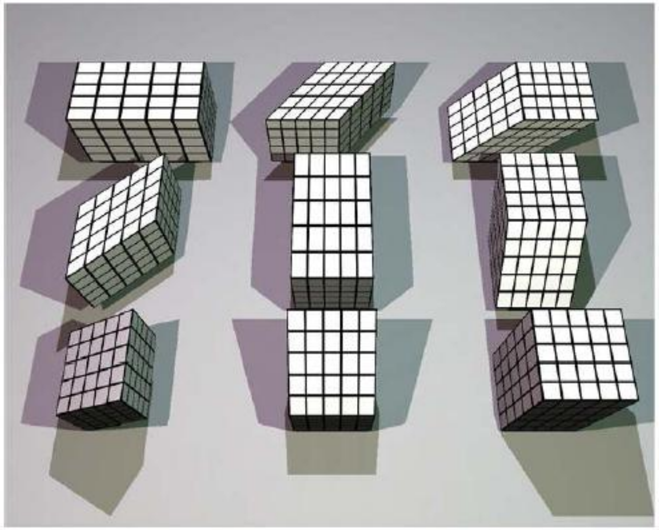
- $\alpha = [0..1]$ - simulates stiffness
- difference is the way the internal elastic forces are treated

# Extensions: Rigid Body Dynamics

- $\alpha = 1$
- points are moved to the goal positions $g_i$ exactly at each time step
- positions represent a rotated and translated version of the initial shape

# Extensions: Linear Deformations

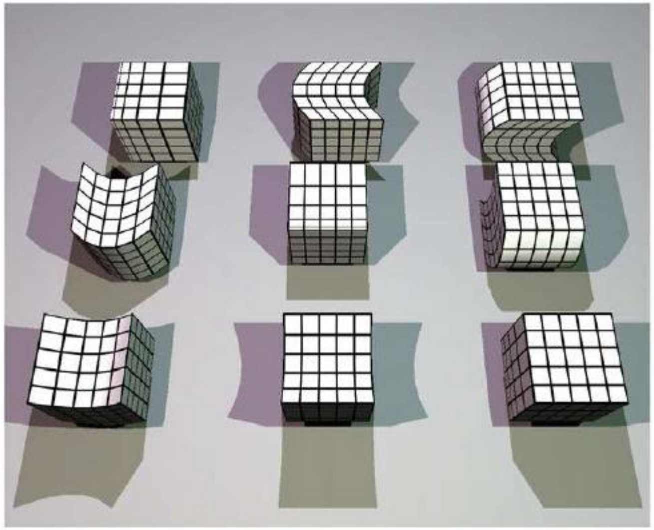
- A – matrix of the best linear transformation to match the actual shape in the least squares sense
- $g_i = \beta A + (1 - \beta)R$
- instead of using just R



# EXTENSIONS: QUADRATIC DEFORMATIONS

- quadratic transformation: $\mathbf{g}_i = [\mathbf{A}\ \mathbf{Q}\ \mathbf{M}]\tilde{\mathbf{q}}_i$
- optimal quadratic transformation:

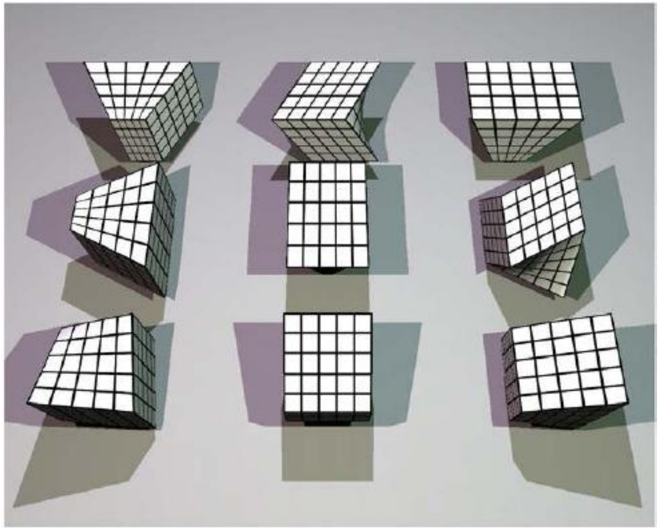
$$\tilde{\mathbf{A}} = \left(\sum_i m_i \mathbf{p}_i \tilde{\mathbf{q}}_i^T\right)\left(\sum_i m_i \tilde{\mathbf{q}}_i \tilde{\mathbf{q}}_i^T\right)^{-1} = \tilde{\mathbf{A}}_{pq}\tilde{\mathbf{A}}_{qq}$$



# Extensions: Cluster Based Deformation

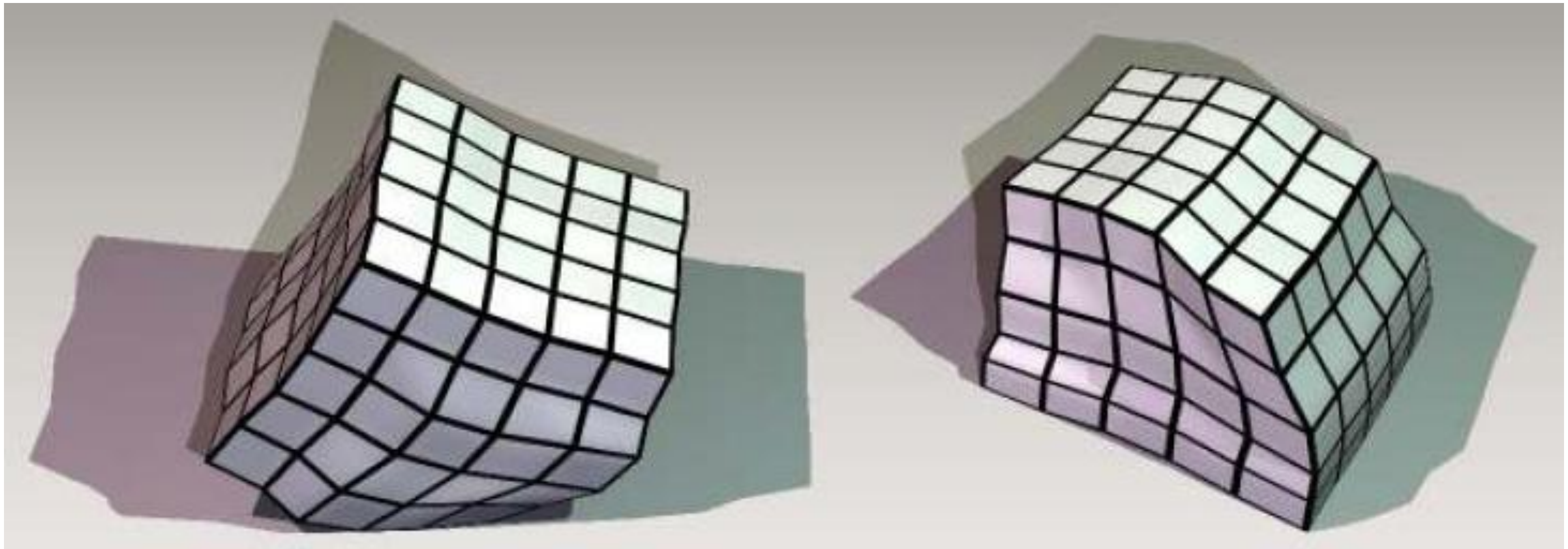
- extend the range of motion
- regularly subdivide the space around a given surface mesh into overlapping cubical regions

$$\Delta \mathbf{v}_i = \alpha \frac{\mathbf{g}_i^c(t) - \mathbf{x}_i(t)}{h}$$



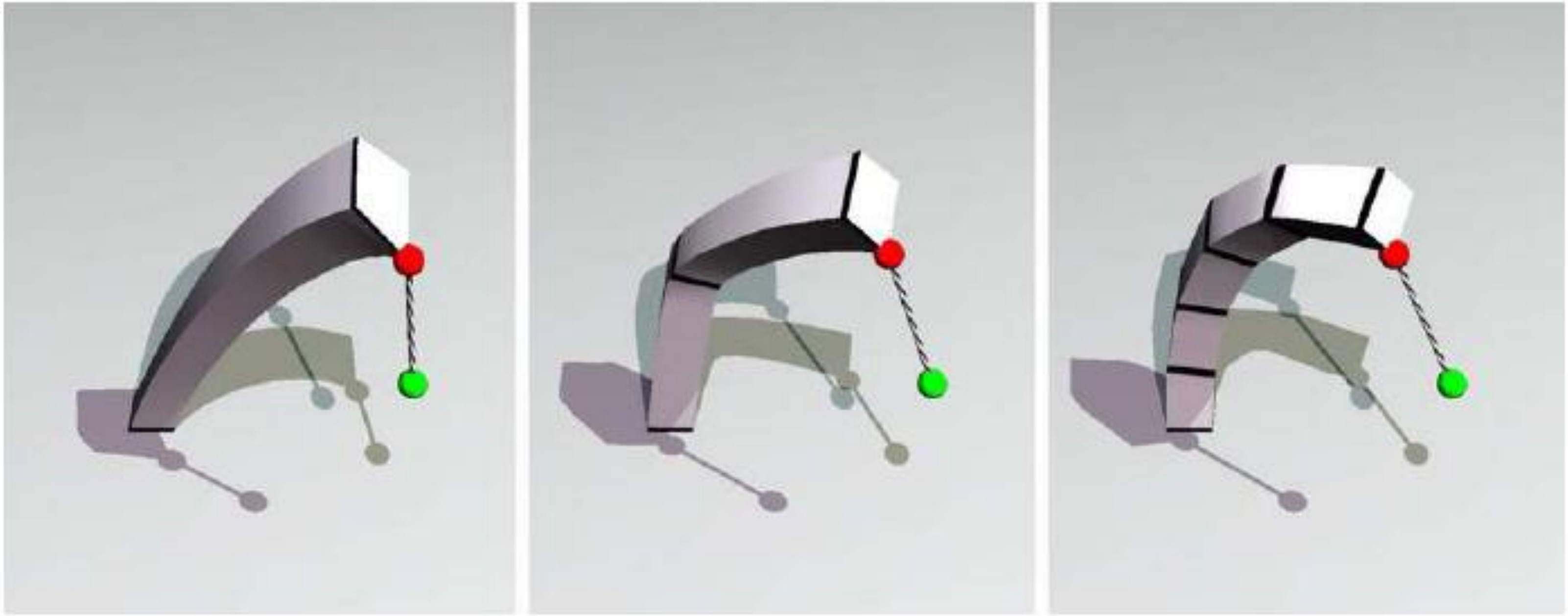
# Extensions: Plasticity

$$\mathbf{S}^p \leftarrow [\mathbf{I} + hc_{\text{creep}}(\mathbf{S} - \mathbf{I})]\mathbf{S}^p$$

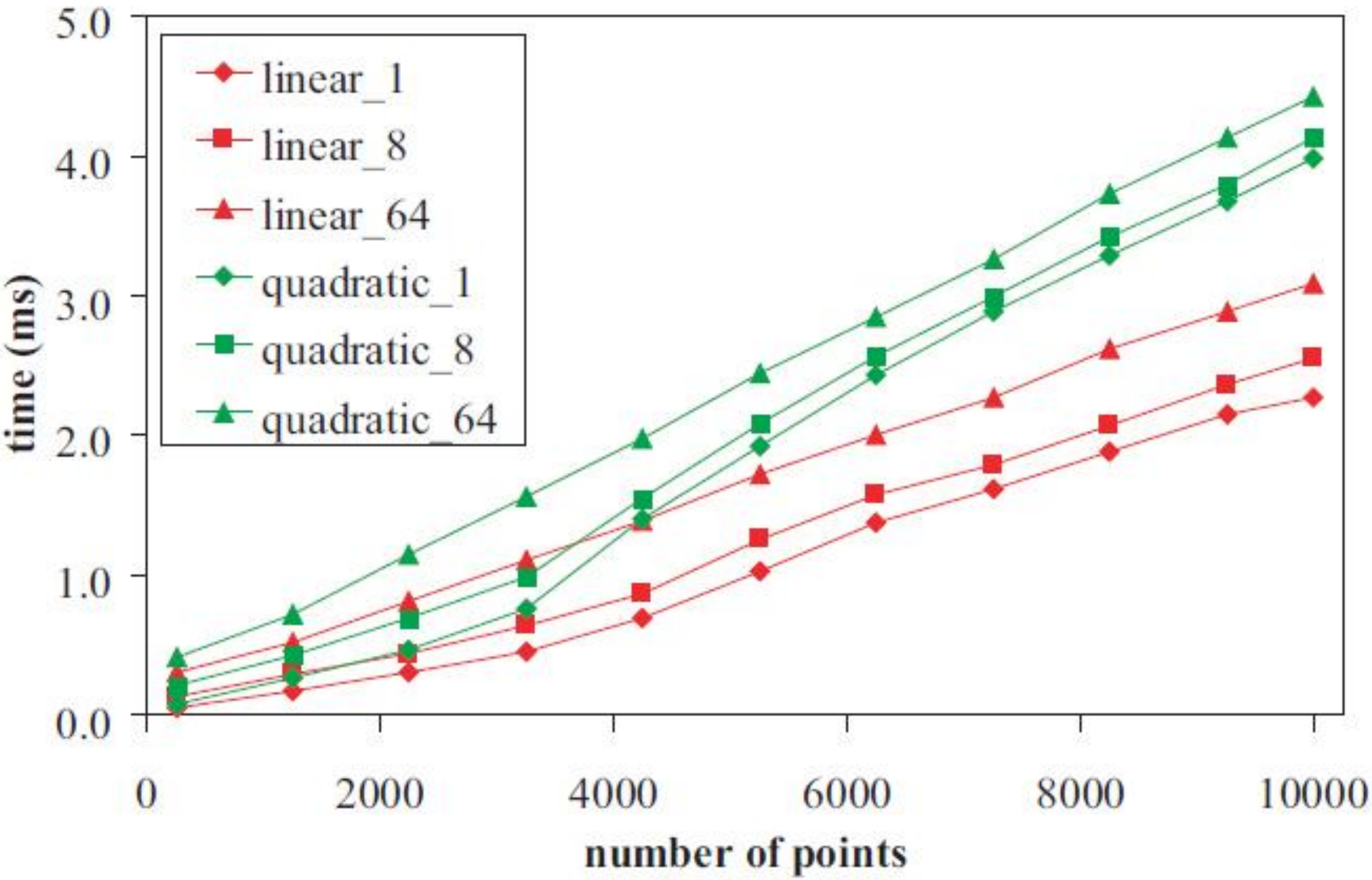


# RESULTS

- test on PC Pentium 4, 3.2 GHz
- Cluster based deformation



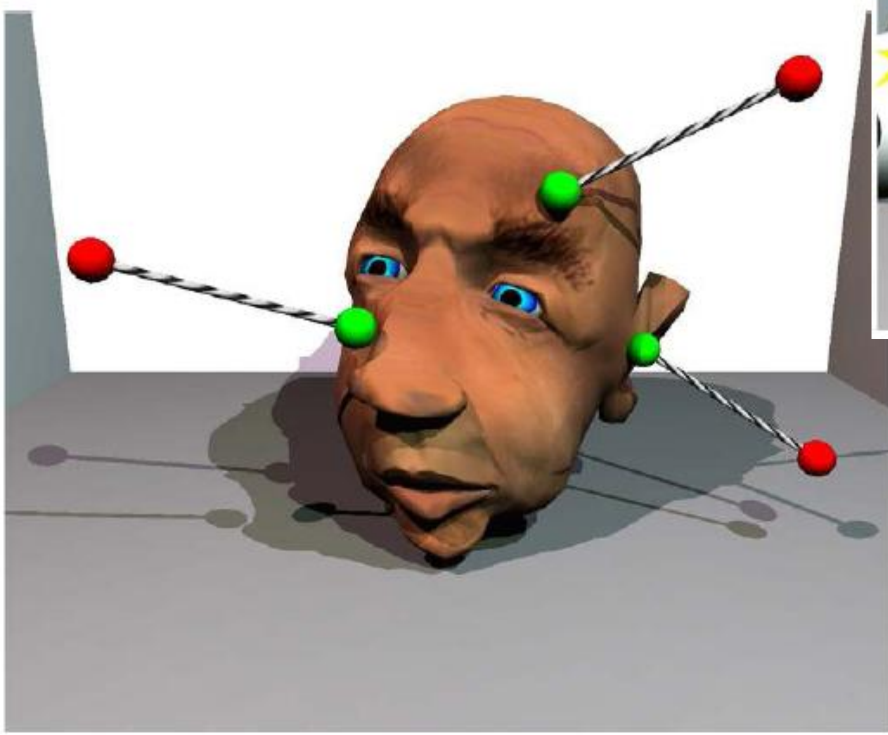
# RESULTS: PERFORMANCE



# Results: Complex simulation scenarios

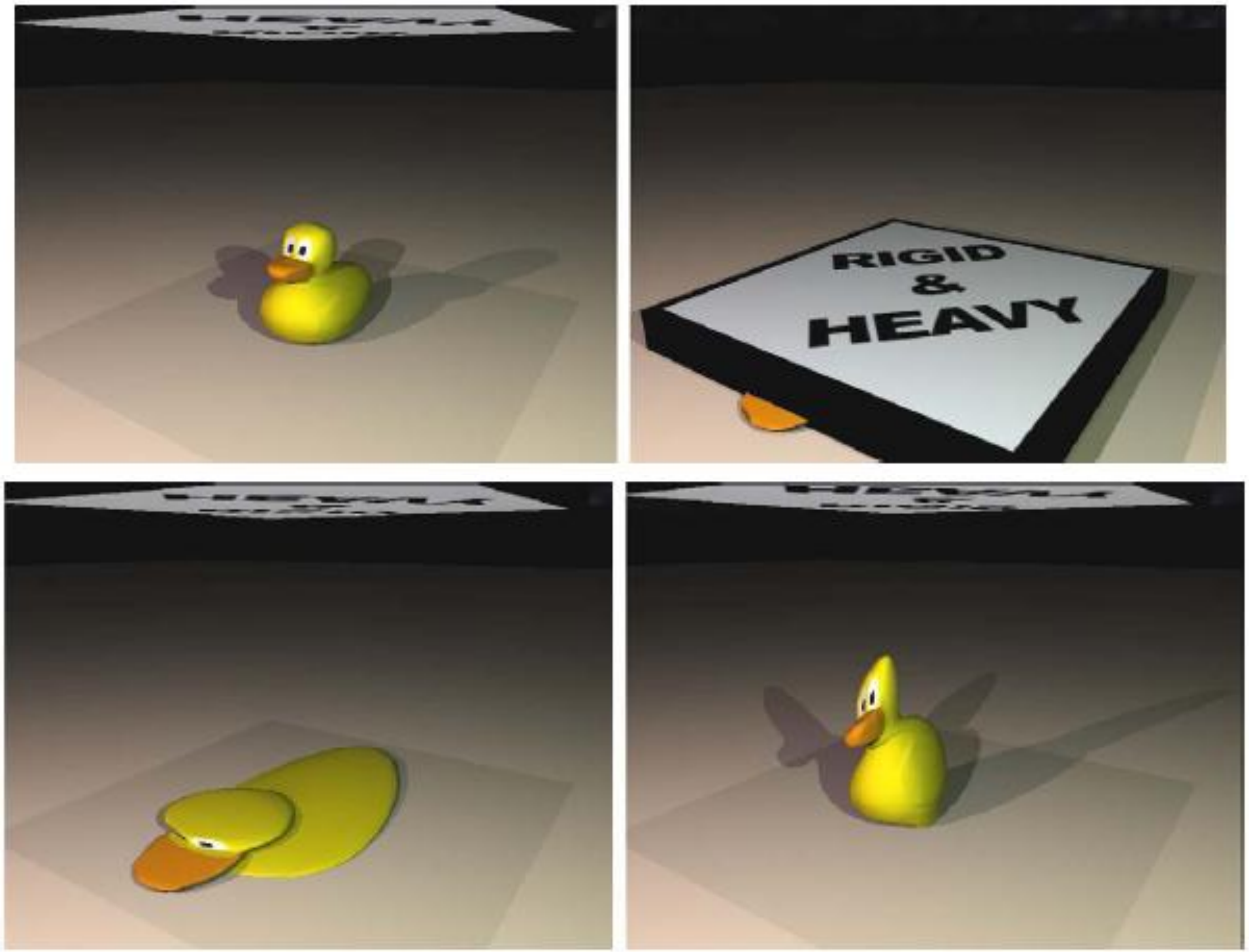
- 384 objects, 2,448 clusters, 55,200 points
- quadratic shape matching take 0.008 and 0.096 milliseconds per frame

# Results: Interactivity



8 clusters and 66 points,
6,460 + 2,000 faces

# Results: Stability



# THE END

Thank you for your attention

Please don’t be shy to ask any question