

The control of physically-based animation

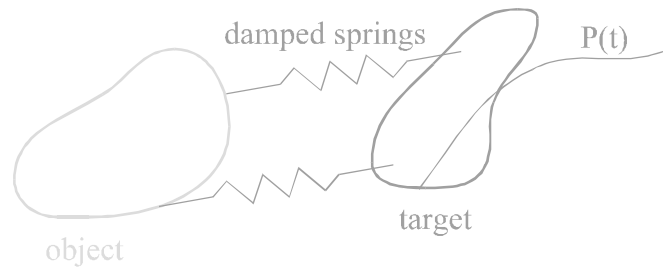
- Forward and inverse dynamics
- Tracking
- Space-time constraints
- Controllers

Forward and inverse dynamics

- Forward dynamics
 - Given forces and torques, compute motions
- Inverse dynamics
 - Given motion, compute forces and torques
- Hybrid dynamics
 - At each DOF, set either force(torque) or motion
 - Deduce the complementary value

Tracking

- A dynamic objects follows an invisible target



- Difficulty: coordination of several objects

Space-time constraints

- Principle
 - consider a whole time interval
 - define the desired animation as a set of constraints over time
 - define a cost function
 - perform a constrained minimization over the time interval



Space-time constraints: example

- One particle in one dimension
- Go from A to B using a force f in a given time



$$x(t) = \mathbf{x} = (x_1, x_2, \dots, x_n)$$

$$f(t) = \mathbf{f} = (f_1, f_2, \dots, f_n)$$

- Be as lazy as possible

Dynamic relations

- Approximation of the derivatives

$$\dot{x}_i = \frac{x_i - x_{i-1}}{h}$$

$$\ddot{x}_i = \frac{x_{i+1} - 2x_i + x_{i-1}}{h^2}$$

- Newton's law

$$m \frac{x_{i+1} - 2x_i + x_{i-1}}{h^2} = f_i$$

Constrained optimization

- Control variables: f_1, f_2, \dots, f_n
- Constraints:

$$c_a = x_1 - a = 0$$

$$c_b = x_n - b = 0$$

$$c_i = m \frac{x_{i+1} - 2x_i + x_{i-1}}{h^2} - f_i = 0$$
- Minimize

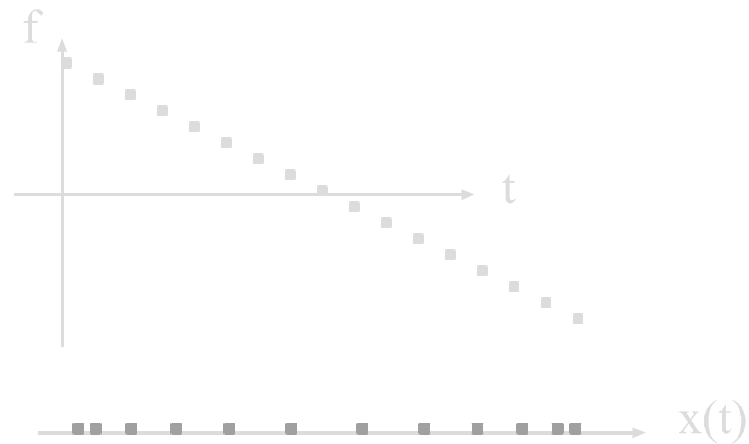
$$R = \sum_{i=1}^n f_i^2$$

Optimization method

- Constraint Jacobian $\mathbf{J} = \left[\frac{\partial c_i}{\partial f_j} \right]$
- Hessian matrix of the cost function $\mathbf{H} = \left[\frac{\partial^2 R}{\partial f_i \partial f_j} \right]$
- Start from an initial guess \mathbf{f}
- Iterate:
 - minimize cost $\mathbf{H}\hat{\mathbf{f}} = -\frac{\partial R}{\partial \mathbf{f}}$
 - apply constraints $\mathbf{J}(\hat{\mathbf{f}} + \tilde{\mathbf{f}}) = -\mathbf{c}$
 - update solution $\mathbf{f}_+ = \hat{\mathbf{f}} + \tilde{\mathbf{f}}$

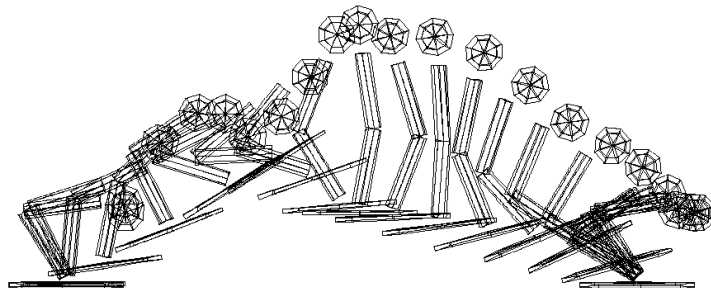
Result

- The goal is met using minimal forces



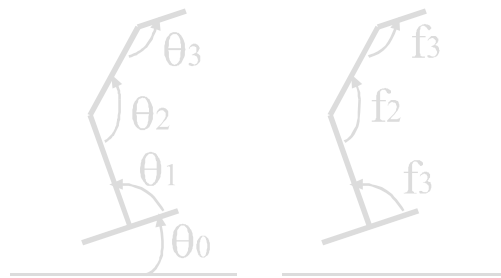
Example: Luxo jump

- A desk lamp must jump to a given point



Model of Luxo

- DOF and actuators



Other kind of space-time optimization

- Model a trajectory using control points
 - Minimize necessary forces
 - the variables are the control points
 - forces are computed using inverse dynamics
- ⇒ Realistic transitions between given motions

Utility of space-time constraints

- Optimize motions using given constraints and criteria
 - realistic animations
 - transitions between precomputed motions
- Problem: big optimizations
 - expensive
 - may need expert users
- We need reusable control laws

Controllers

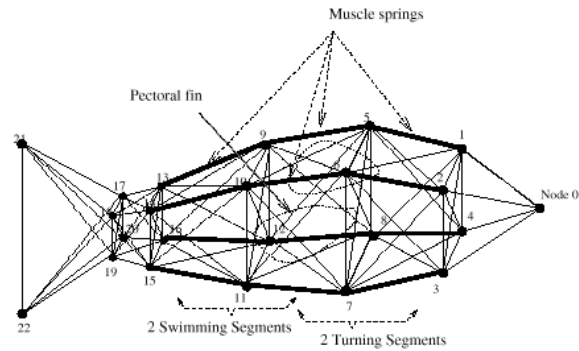
- Define forces as a function of the current state



- Tune the function parameters for a given task

Example of controller: motion of a fish

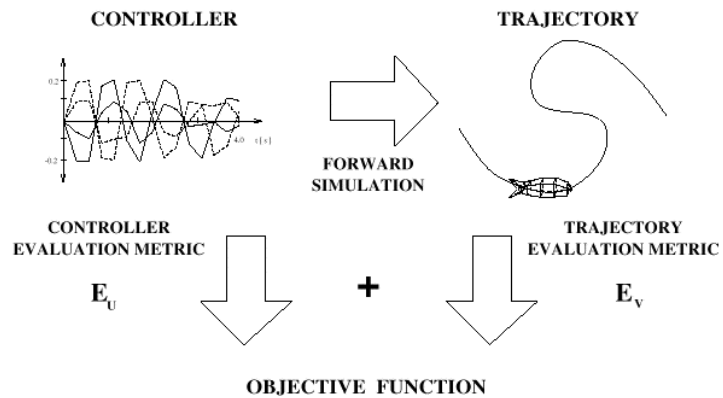
- Fish model



- Dynamics: motion in function of muscle forces

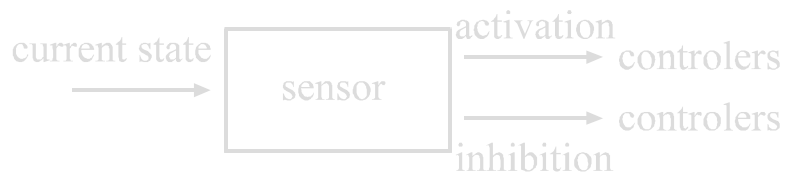
Fish controller

- Optimize controllers for given trajectories and criteria



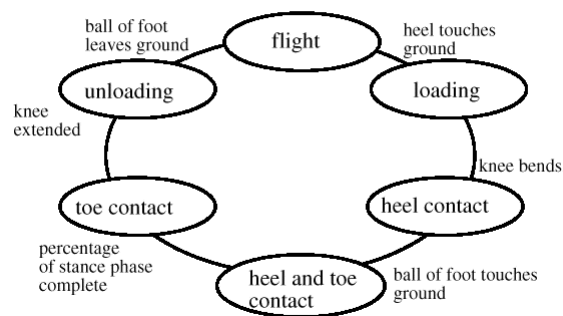
Generic control

- Parametrizable controllers can be reused. Ex:
 - move forward(velocity)
 - turn right(velocity,angle)
- Controllers can be triggered procedurally or using sensors



Control graphs

- Different control laws can be applied according to the current state
- Ex: human running



Complex control

- Complex tasks involve the combination of several sensors
- Sensor-actuator networks

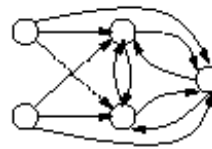


FIGURE 5. Topology of a sensor-actuator network

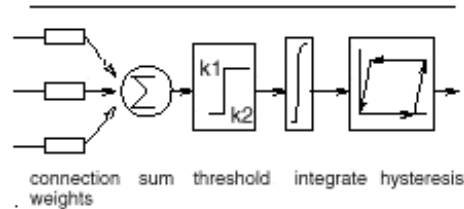


FIGURE 6. Function of a SAN node

Utility of dynamics for computer animation

- + Realistic motion
- Much computation
- Control is difficult

References

- Witkin, Kass, *Spacetime constraints*, SIGGRAPH'98
- van de Panne, Fiume, *Sensor-actuator networks*, SIGGRAPH'93
- Liu, Gortler, Cohen, *Hierarchical spacetime control*, SIGGRAPH'94
- Grzeszczuk, Terzopoulos, *Automated learning of muscle-actuated locomotion through control abstraction*, SIGGRAPH'95
- Hodgins, Wooten, Brogan, O'Brien, *Animating human athletics*, SIGGRAPH'95