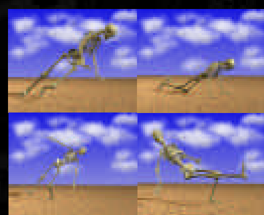
Composable Controllers for Physics-based Character Animation

Petros Faloutsos
Michiel van de Panne
Demetri Terzopoulos

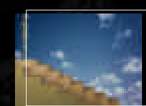
University of Toronto
Dept. Of
Computer Science



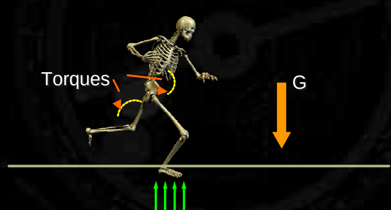
Animation Techniques

Making graphical characters move

- State-of-the-art:
  predominantly kinematic
- Reality:
  physics + control

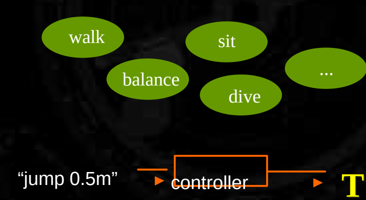


Physics



Control

Divide and conquer



Questions

What goes in a controller?

How do we arbitrate between controllers?

- Controllers model their own abilities

Related Work

Controller Design

- Athletic motions [Hodgins 95]
- Leaping, Tumbling, Landing, Balancing [Wooten 98]
- Walking controllers [Laszlo 96]

Controller Composition

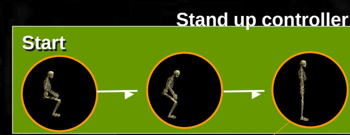
- Juggling [Burridge 99]

Designing Controllers

Difficult !

- How do people move?
  - *Study biomechanics*
  - *Use intuition*
- Helpful Techniques
  - *Manual trial and error*
  - *Automatic refinement*
- Strategy: Basic to advanced

Finite State Machine



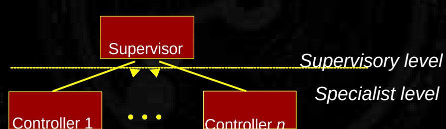
$$q_1 \rightarrow t_1 = k_s^1(q_1 - q_{1,desired}) - k_d^1\dot{q}_1$$

$$\vdots$$

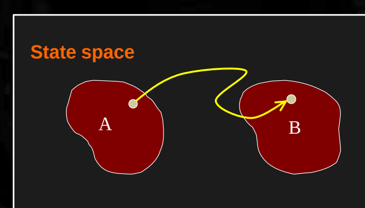
$$q_n \rightarrow t_n = k_s^n(q_n - q_{n,desired}) - k_d^n\dot{q}_n$$

Controller Composition

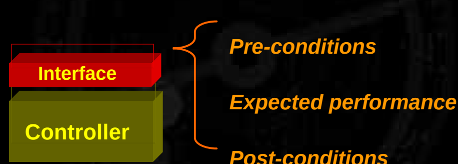
Two Level Control Scheme



Intuition

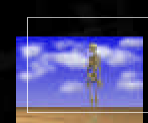


"Composable" Controllers



Pre-conditions

Know When to Operate



- Initial state: $\mathbf{q}_i$
- Environment: C_e

$$P = P(R(\mathbf{q}_i), C_e)$$

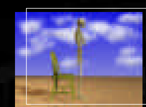
Expected Performance

Know When to Quit

- Unexpected situations
- Failure or early success

$$? = C(R(\mathbf{q}(t), C_e(t))$$

Post-conditions

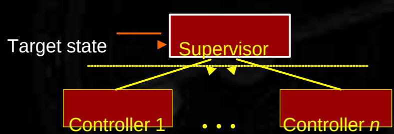


Know the Target or Goal

- Target state, $\mathbf{q}_o$
- Environment, C_e

$$O = C(R(\mathbf{q}_o), C_e)$$

Controller Arbitration

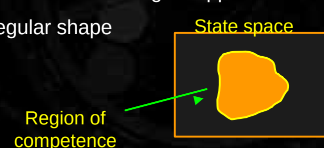


- How to arbitrate ?
- When to arbitrate ?

Determining Pre-conditions

Difficult !

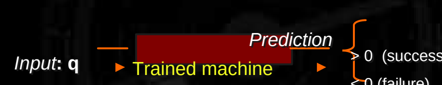
- Multidimensional region approximation
- Irregular shape



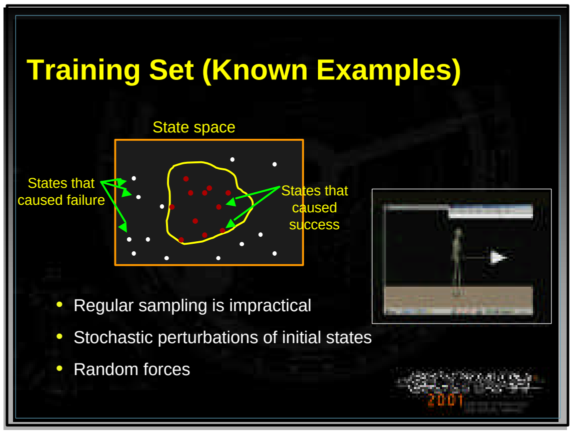
Two Solutions

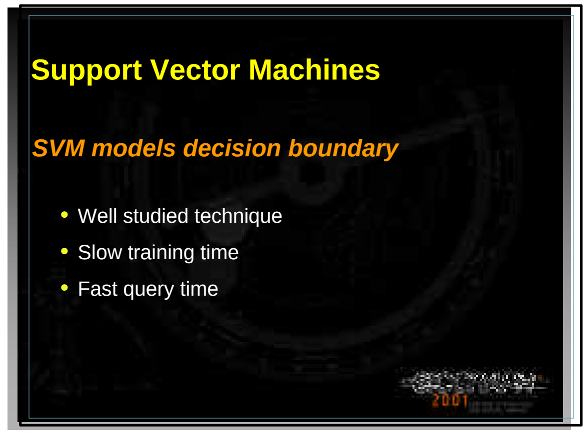
- Analytic or manual
- Machine learning

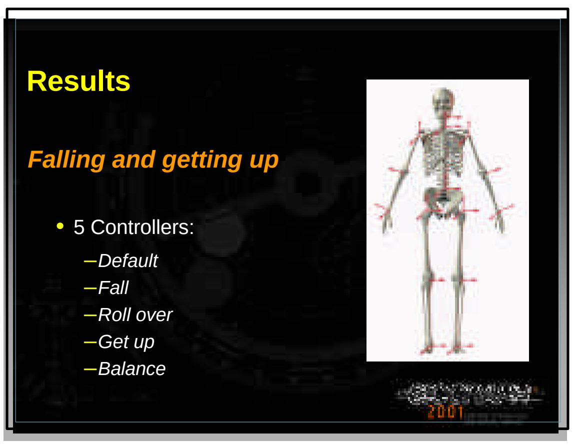
Learning of Preconditions

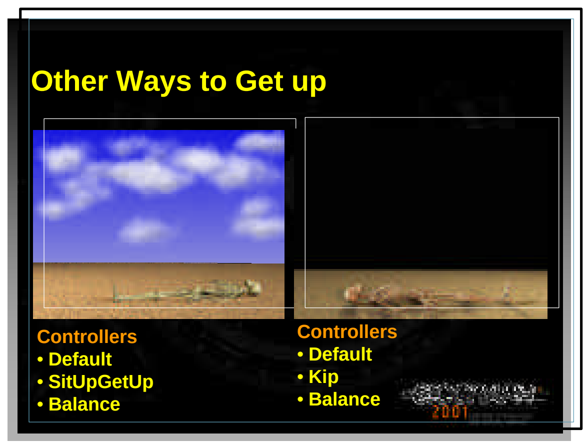


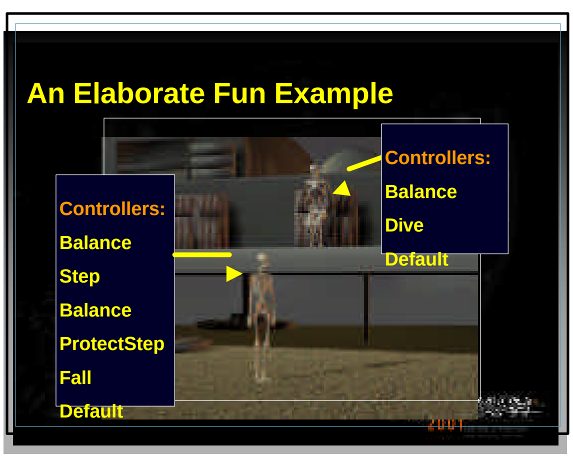
classification problem

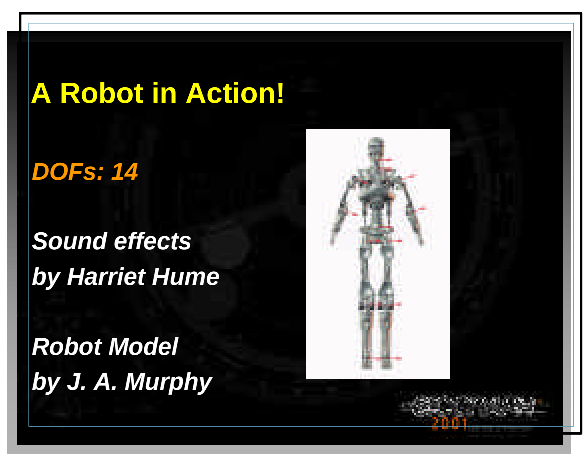

Training Set (Known Examples)
State space
States that caused failure
States that caused success
Regular sampling is impractical
Stochastic perturbations of initial states
Random forces
2001


Support Vector Machines
SVM models decision boundary
Well studied technique
Slow training time
Fast query time
2001


Results
Falling and getting up
5 Controllers:
–Default
–Fall
–Roll over
–Get up
–Balance
2001


Other Ways to Get up
Controllers
• Default
• SitUpGetUp
• Balance
Controllers
• Default
• Kip
• Balance
2001


An Elaborate Fun Example
Controllers:
Balance
Step
Balance
ProtectStep
Fall
Default
Controllers:
Balance
Dive
Default
2001


A Robot in Action!
DOFs: 14
Sound effects
by Harriet Hume
Robot Model
by J. A. Murphy
2001

Conclusions

Control Framework

- Controllers model their own competency
- Exchangeable and composable controllers
- On-the-fly controller selection

DANCE software system (with Victor Ng)

- Unifying platform for collaborative research
- Publicly available

Future Work

Towards real-time

Cooperative controllers

Planning

Acknowledgements

Victor Ng-Thow-Hing
Joe Laszlo
Michael Neff
Glenn Tsang
Meng Sun

Anastasia Bezerianos
David Mould
Chris Trendal
Harriet Hume

Questions?

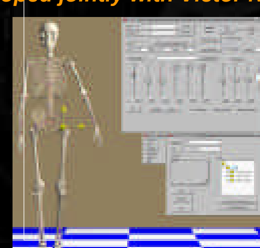
Questions ?

Overview

Introduction
Controller design
Controller composition
SVM learning of controller competency
DANCE software system
Conclusion

DANCE
(Dynamic Animation and Control Environment)

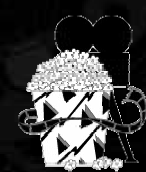
Developed jointly with Victor Ng-Thow-Hing



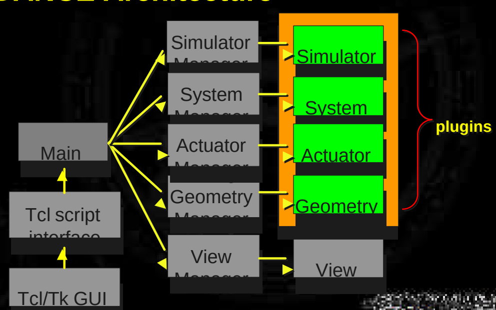
Features

- Common platform for animation research
- Component abstraction for physics-based animation environments
- Communication between abstract modules
- Innovative plug-in architecture

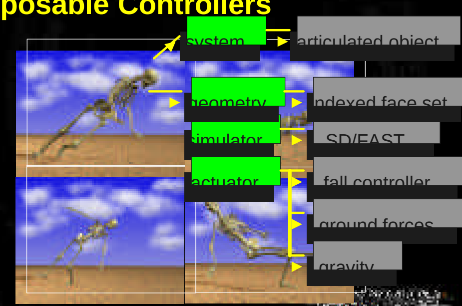
DANCE is Powerful



DANCE Architecture

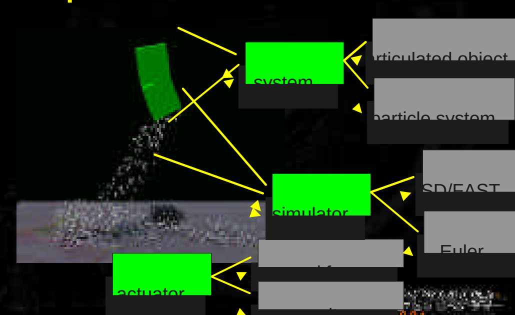


Example I: Composable Controllers



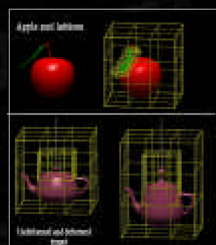
Example II: Muscles (Victor Ng)

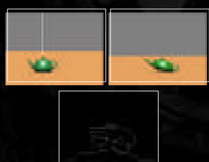
Example III: Salt Shaker



Example IV: Dynamic Free-Form Deformations [IEEE TVCG'97]

Lagrangian dynamics
Hierarchical deformations





Dynamic Free-form Deformations

Equations of motion:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{J}^T\mathbf{F}_{ext} + \mathbf{Q}_c + \frac{1}{2}\dot{\mathbf{q}}^T\frac{\partial \mathbf{M}}{\partial \mathbf{q}}\dot{\mathbf{q}} - \left(\frac{\partial \mathbf{M}}{\partial t}\right)\dot{\mathbf{q}}$$

State variables $\mathbf{q} = [\mathbf{T}\ \mathbf{R}\ ?_{global}\ \ ?_{l,1} \cdots ?_{l,n}]$

$$P = \mathbf{T} + \mathbf{R}\overbrace{\sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3}\left(\sum_{n=1}^{D_g} g_n^g \mathbf{d}_{gijk}^n + \mathbf{L}_{gijk}\right)B(s_g,i)B(t_g,j)B(u_g,k)}^{\text{deformed position}}$$

$$\begin{bmatrix} s_g \\ t_g \\ u_g \end{bmatrix} = \mathbf{a}\sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3}\left(\sum_{n=1}^{D_g} g_n^l \mathbf{d}_{lijk}^n + \mathbf{L}_{lijk}\right)B(s_l,i)B(t_l,j)B(u_l,k)$$

Who Wants to DANCE ?

Do not re-invent the wheel!

- Researchers in robotics, graphics, biomechanics
- Students (classroom tool)
- Game programmers

DANCE Saves a Lot of Time

	Without	With
Base system: ***command shell, interface component classes***	*4 months*	*0*
Articulated figure + Simulator: ***data structures, in/out, rendering***	*5 months*	*0*
Actuators: ***Ground, Playback, Gravity, Collision***	*2 months*	*0*
Misc: ***utility functions, optimization, debugging***	*1 month*	*0*
Getting familiar with DANCE	*0*	*3 hours*
	13 months	***3 hours***

DANCE Is Already in Use

Dept. of Computer Science, University of Toronto

- Interactive virtual puppetry
- Interactive animation of characters

Dept. of Anatomy, University of Toronto
Human Simulation Group, Stanford University

DANCE code is free for non-commercial use at

http://www.dgp.toronto.edu/DGP/software/dance

Overview

Introduction
Controller design
Controller composition
SVM learning of controller competency
DANCE software system
Conclusion

Contributions

Control System Framework for Autonomous Dynamic Characters

- Extensible
- Dynamic Composition
- Controllers learn their competency

DANCE software system (with Victor Ng)

- Unifying platform for collaborative research

Related Work

Controller Design

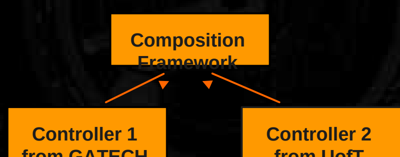
- Athletic motions [Hodgins 95]
- Walking controllers [Laszlo 96]
- Falling [Smeesters 99]

Controller Composition

- Leaping, Tumbling, Landing, Balancing [Wooten 98]
- Juggling [Burridge 99]

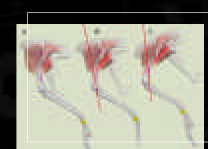
Composition: Another Benefit

Harness independent results



Biomechanics

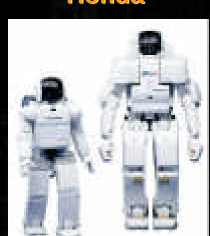
Modeling human and other animal bodies (Victor's muscle model)



Understanding and simulating motor control

Potential Applications to Robotics

Honda

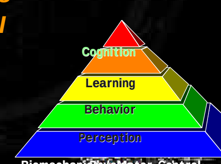


Sony

MIT Leg Lab

Future Work

Efficient simulation
Multiple controllers
Motor control and planning
Behavior animation and AI



Acknowledgements

Victor Ng-Thow-Hing

Joe Laszlo

Michael Neff

DGP lab group at the University of Toronto

- **Glenn Tsang, Meng Sun, Anastasia Bezerianos, David Mould, Chris Trendal**

Harriet Hume (Sound effects)

Publications/Software

Graphics

- Composable Controllers [ACM SIGGRAPH 2001]
- Animation of Deformable Characters [IEEE TVCG 1997]
- DANCE Software System [ACM SIGGRAPH 2000]

Robotics

- Dynamic Articulated Characters [ICRA 2000]

Visualization

- Modeling Real Computer Networks [ACM SIGCOMM 1999]
- Bibliographic Data Visualization [CASCON 1996]

Software

- DANCE www.dgp.toronto.edu/dgp/software/dance
- Graphics Engine www.thelivingletters.com

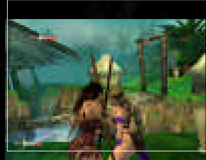
Thank You!

Character Animation in the Film Industry

Disney's Jurassic Park

Pixar's Toy Story

Character Animation in the Game Industry



Xena by Electronic Arts

NHL by Electronic Arts

Shenmue by AM2 and SEGA

Numerical Solution

Discrete time approximation

Given $\mathbf{q}(t)$ and $\dot{\mathbf{q}}(t)$

Compute $\mathbf{M}, \mathbf{b}$

Solve $\mathbf{M}\ddot{\mathbf{q}} = \mathbf{b}$ for $\ddot{\mathbf{q}}$

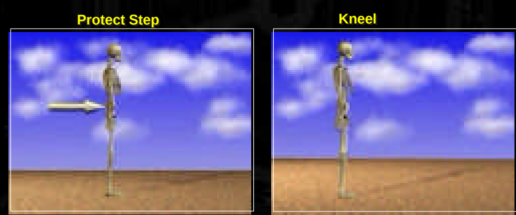
Intergrate

$$\dot{\mathbf{q}}(t+\Delta t) = \dot{\mathbf{q}}(t) + \Delta t \ddot{\mathbf{q}}(t)$$

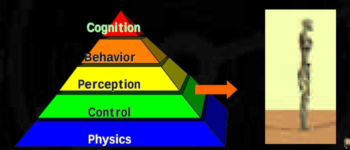
$$\mathbf{q}(t+\Delta t) = \mathbf{q}(t) + \Delta t \dot{\mathbf{q}}(t) + \frac{\Delta t^2}{2}\ddot{\mathbf{q}}(t)$$

Can be computationally expensive!

Examples



Long-Range Goal: Intelligent Virtual Characters



First step: Self-Animating Characters

- Rich repertoire of motor skills
- Capable of autonomous, reactive movement

Contributions

- Control System Framework for Autonomous Dynamic Characters
  - *Dynamic composition*
  - *Extensible*
  - *Controllers learn their competency*
- Runs on top of DANCE (DANCE jointly developed with Victor Ng-Thow-Hing)
- Share and re-use results

Overview

Introduction
Controller design
Controller composition
SVM learning of controller competency
Conclusion

Overview

Introduction
Controller design
Controller composition
SVM learning of controller competency
Conclusion

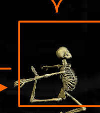
Applying Newtonian Mechanics

Nonlinear 2nd-order differential eqs of motion

$$\mathbf{M}(\mathbf{q},t)\ddot{\mathbf{q}}(t)+C(\mathbf{q},\dot{\mathbf{q}},t)=\sum_i \mathbf{J}_T^T\mathbf{f}_i+\sum_l \mathbf{J}_R^T\mathbf{t}_{ext,l}+\sum_k \mathbf{J}_R^T\mathbf{t}_{j,k}$$

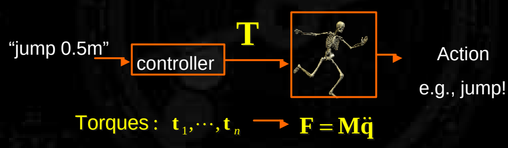
Inertial Forces, Gyroscopic Forces, External Forces, External Torques, Joint Torques

Forces and Torques $[\mathbf{f}\ \mathbf{t}_{ext}\ \mathbf{t}_j]$



State $[\mathbf{q}\ \dot{\mathbf{q}}]$

What is a "Controller"?



Overview

Introduction

Controller design

Controller composition

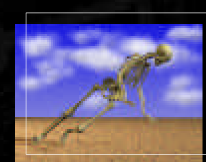
SVM learning of controller competency

Conclusion

Composite Fall Controller

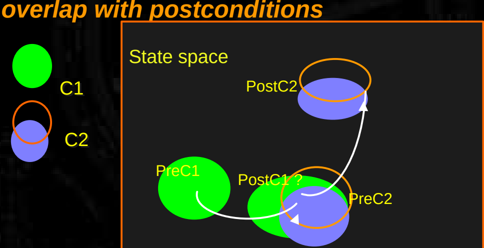
Standing ® Step ® Fall

- Fall Controller
  - *Pre-conditions:*
    $\mathbf{c} \notin \{SP\}, \ \dot{\mathbf{c}} > \mathbf{a}$
  - *Expected performance:*
    Contact with ground in 2 seconds
    $\dot{\mathbf{c}} > \mathbf{a}'$
  - *Post-conditions:*
    $\dot{\mathbf{c}} \sim 0$



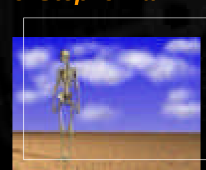
Transitions Between Controllers

Transitions happen when preconditions overlap with postconditions

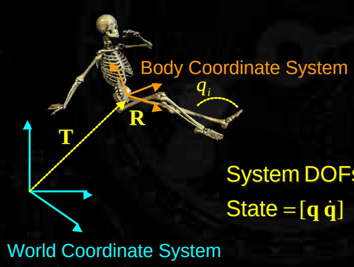


Composite Fall Controller

Standing ® Step ® Fall



Notation



System DOFs $\mathbf{q} = [\mathbf{T}\ \mathbf{R}\ ?_1 \cdots ?_n]$

State $= [\mathbf{q}\ \dot{\mathbf{q}}]$

Overview

Introduction

Controller design

Controller composition

SVM learning of controller competency

DANCE software system

Conclusion

SVM Theory

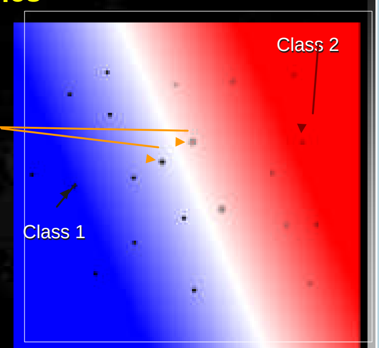
Method for fitting functions to training data

- Introduced by Vapnik in 1979
- *Decision function* $f(\mathbf{x}, a)$ classifies observation **x**
  (*a: Lagrange multipliers)*
- Solve a quadratic programming problem for the a
- Training data for which $a > 0$ are *the support vectors*
- *Kernel functions* generalize linear SVM theory to nonlinear cases (Gaussian (RBF), polynomial, sinusoidal, linear…)
- Extensions of SVM theory:
  prior knowledge, fast classification

SVM Examples in 2D

Linear

Support vectors

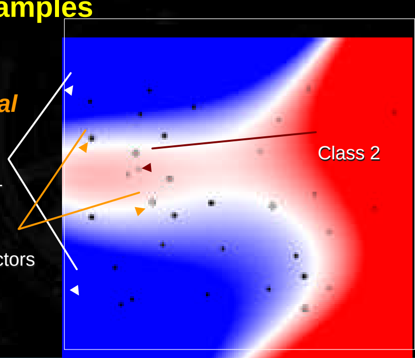


SVM Examples in 2D

Polynomial

Class 1

Support vectors

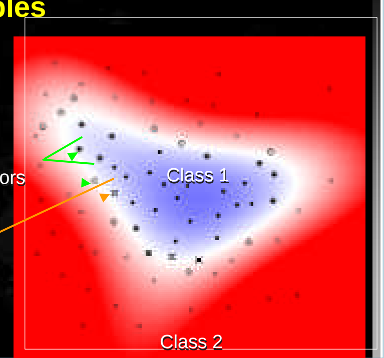


SVM Examples in 2D

Gaussian

Support vectors

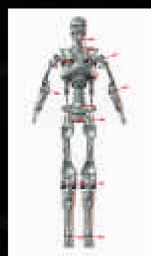
Misclassified



A Robot in Action!

DOFs: 14

Sound effects
by Harriet Hume



The SVM Method Works Well

Controller	Training set size	Test set size	Nearest Neighbor	SVM
BentToStand	6,926	14,272	98.05%	99.77%
StandIn Place	17,317	20,393	83.63%	87.67%
Walk	11,020	8,658	92.78%	97.73%

SVMs and Random Element

Controller	Succes no noise %	#Support vectors	Success % with noise	#Support vectors
Dstance to Stand	87.29	1604	86.68	1660
Walk	96.73	738	96.17	862
Crouch to Stand	99.77	41	99.73	54

SVMs and Noise

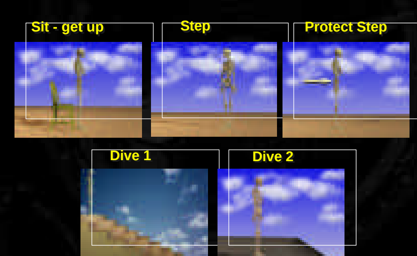
Noise	SVM	#Sup. Vecs	NN
0.00	87.29	1604	80.97
0.01	87.48	1625	81.05
0.10	87.23	1811	80.87
1.00	81.86	3719	65.70

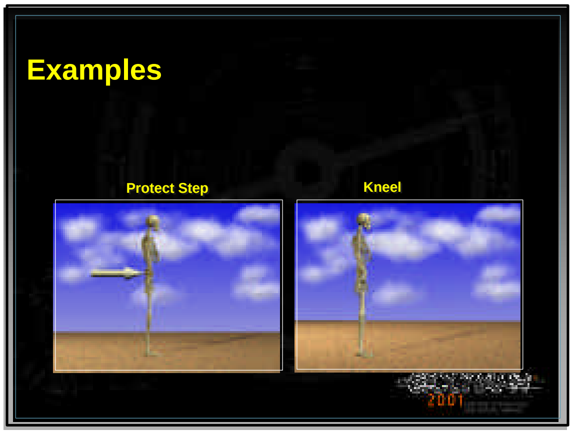
DStanceToStand

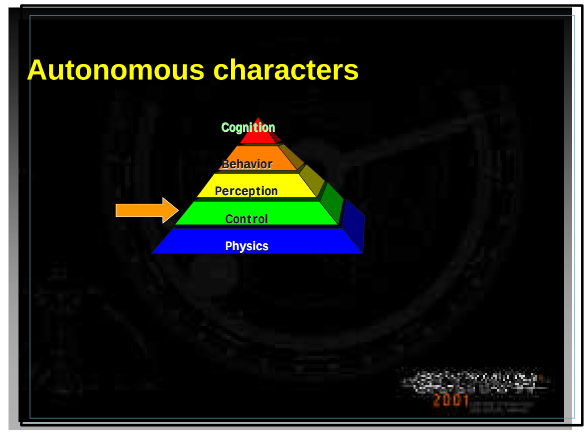
Overview

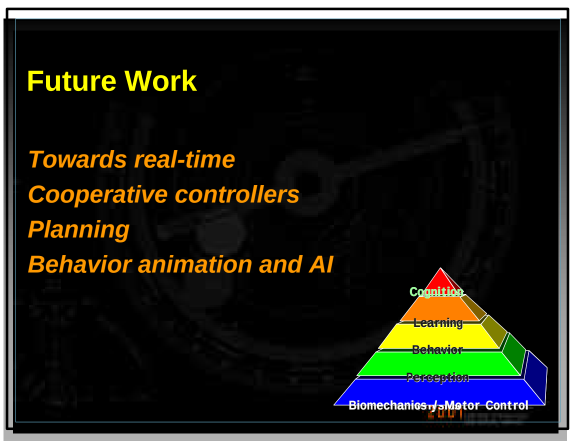
Introduction
Controller design
Controller composition
SVM learning of controller competency
Conclusion

Simple Composition Examples




Examples
Protect Step
Kneel
2001


Autonomous characters
Cognition
Behavior
Perception
Control
Physics
2001


Future Work
Towards real-time
Cooperative controllers
Planning
Behavior animation and AI
Cognition
Learning
Behavior
Perception
Biomechanics / Motor Control