Physics and Animation

Niloy Mitra  Iasonas Kokkinos  Federico Monti  Emanuele Rodolà  Michael Bronstein  Or Litany  Leonidas Guibas

UCL  UCL  USI Lugano  La Sapienza  Imperial College USI Lugano  Stanford University Facebook  Stanford University

http://geometry.cs.ucl.ac.uk/dl_for_CG/
## Timetable

<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
<th>Lecturers</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00-10:30</td>
<td><strong>Introduction</strong></td>
<td>Niloy X</td>
</tr>
<tr>
<td></td>
<td><strong>Machine Learning Basics</strong></td>
<td>Federico X</td>
</tr>
<tr>
<td></td>
<td><strong>Neural Network Basics</strong></td>
<td>Iasonas X</td>
</tr>
<tr>
<td></td>
<td><strong>Alternatives to Direct Supervision (GANs)</strong></td>
<td>Emanuele X</td>
</tr>
<tr>
<td>~9:05</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~9:35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~11:00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>~11:45</td>
<td><strong>Image Domain</strong></td>
<td></td>
</tr>
<tr>
<td>~13:30</td>
<td><strong>3D Domains (extrinsic)</strong></td>
<td></td>
</tr>
<tr>
<td>~14:15</td>
<td><strong>3D Domains (intrinsic)</strong></td>
<td></td>
</tr>
<tr>
<td>~16:00</td>
<td><strong>Physics and Animation</strong></td>
<td></td>
</tr>
<tr>
<td>~16:45</td>
<td><strong>Discussion</strong></td>
<td></td>
</tr>
</tbody>
</table>

Sessions: A. 9:00-10:30 (coffee) B. 11:00-12:30 [LUNCH] C. 13:30-15:00 (coffee) D. 15:30-17:00
NN Cheatsheet
NN Cheatsheet

- Data, data, data
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
• Use existing networks to start, if possible
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
• Use existing networks to start, if possible
• Overfit, ‘reproducible’, backprop (check if possible)
NN Cheatsheet

- Data, data, data
- Setup evaluation, benchmark, loss measures, baselines
- Initialize well, visualize intermediate results
- Use existing networks to start, if possible
- Overfit, ‘reproducible’, backprop (check if possible)
- ADAM
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
• Use existing networks to start, if possible
• Overfit, ‘reproducible’, backprop (check if possible)
• ADAM
• Change one block/concept at a time
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
• Use existing networks to start, if possible
• Overfit, ‘reproducible’, backprop (check if possible)
• ADAM
• Change one block/concept at a time
• Regularize
  (e.g., latent representation, Spectral basis, image formation module)
NN Cheatsheet

• Data, data, data
• Setup evaluation, benchmark, loss measures, baselines
• Initialize well, visualize intermediate results
• Use existing networks to start, if possible
• Overfit, ‘reproducible’, backprop (check if possible)
• ADAM
• Change one block/concept at a time
• Regularize
  (e.g., latent representation, Spectral basis, image formation module)
• Hyperparameter optimization
Physics-Based Animation

- Leverage *physical models*

Examples:
- Rigid bodies
- Cloth
- Deformable objects
- Fluids

[Many of the following slides thanks to Nils Thuerey]
Physics-Based Animation

• Leverage *physical models*

• Examples:
  • Rigid bodies
  • Cloth
  • Deformable objects
  • Fluids
Physics-Based Animation

• Traditional approach:
Physics-Based Animation

• Traditional approach:

Experiment

Observations / data

Theory

Computation

Deep Learning for CG & Geometry Processing
Physics-Based Animation

• Traditional approach:

- Observations / data
- Model equations
Physics-Based Animation

• Traditional approach:

Experiment
Observations / data

Theory
Model equations

Computation
Discrete representation
Physics-Based Animation

• Traditional approach:

Experiment
Observations / data

Theory
Model equations

Computation
Discrete representation
Physics-Based Animation

• Traditional approach:

Skip Theory with Deep Learning?

Experiment
Observations / data

Theory
Model equations

Computation
Discrete representation
Physics-Based Animation

• Traditional approach:

Observations / data → Model equations → Discrete representation

Skip Theory with Deep Learning?
[No! More on that later…]
Partial Differential Equations

• Typical problem formulation: unknown function $u(x_1, ..., x_n)$

• PDE of the general form:

$$f(x_1, ..., x_n; \frac{\partial u}{\partial x_1}, ..., \frac{\partial u}{\partial x_n}; \frac{\partial^2 u}{\partial x_1 \partial x_1}, \frac{\partial^2 u}{\partial x_1 \partial x_2}, ...) = 0$$

• Solve in domain $\Omega$, with boundary conditions on boundary $\Gamma$

• Traditionally: discretize & solve numerically. Here: also discretize, but solve with DL...
Methodology 1

• Viewpoints: *holistic* or *partial*

  [partial also meaning “coarse graining” or “sub-grid / up-res”]

• Influences complexity and non-linearity of solution space

• Trade off computation vs accuracy:
  • Target most costly parts of solving
  • Often at the expense of accuracy
Learning to Represent Mechanics
Learning to Represent Mechanics
Methodology 2

• Consider dimensionality & structure of discretization
Methodology 2

• Consider dimensionality & structure of discretization
  • Small & unstructured
    • Fully connected NNs only choice
    • Only if necessary...
Methodology 2

• Consider dimensionality & structure of discretization
  • Small & unstructured
    • Fully connected NNs only choice
    • Only if necessary...
  • Large & structured
    • Employ convolutional NNs
    • Usually well suited
Solving PDEs with DL

• Practical example: *airfoil flow*
  • Given boundary conditions solve stationary flow problem on grid
  • Fully replace traditional solver
  • 2D data, no time dimension
  • I.e., holistic approach with structured data
Solving PDEs with DL

• Data generation
• Large number of pairs: input (BCs) - targets (solutions)
Solving PDEs with DL

- Data generation
- Large number of pairs: input (BCs) - targets (solutions)
Solving PDEs with DL

• Data generation
• Example pair
• Note - boundary conditions (i.e. input fields) are typically constant
• Rasterized airfoil shape present in all three input fields
Solving PDEs with DL

- Data generation
- Example pair
- Note - boundary conditions (i.e. input fields) are typically constant
- Rasterized airfoil shape present in all three input fields
Solving PDEs with DL

- Data generation
- Example pair
- Note - boundary conditions (i.e. input fields) are typically constant
- Rasterized airfoil shape present in all three input fields
Solving PDEs with DL

• U-net NN architecture
Solving PDEs with DL

• U-net NN architecture

Reduce spatial dimensions
Solving PDEs with DL

- U-net NN architecture

Reduce spatial dimensions

Skip connections
Solving PDEs with DL

• U-net NN architecture

Reduce spatial dimensions

Skip connections

Increase spatial dimensions
Solving PDEs with DL

Deep Learning for CG & Geometry Processing
Solving PDEs with DL

• Unet structure highly suitable for PDE solving
Solving PDEs with DL

• Unet structure highly suitable for PDE solving
• Makes boundary condition information available throughout
Solving PDEs with DL

- Unet structure highly suitable for PDE solving
- Makes boundary condition information available throughout
- Crucial for inference of solution
Solving PDEs with DL

- **Training**: 80,000 iterations with ADAM optimizer
- Convolutions with enough data - no dropout necessary
- Learning rate decay stabilizes models
Results

Target

Pressure  Velocity X  Velocity Y
Results

(A) Regular data

Pressure  Velocity X  Velocity Y
Results

![Heatmaps of Pressure, Velocity X, and Velocity Y]

- **Target**
- **(A) Regular data**
- **(B) Dimension less**
Solving PDEs with DL

- Validation and test accuracy for different model sizes
Solving PDEs with DL

• Validation and test accuracy for different model sizes
Solving PDEs with DL

• Validation and test accuracy for different model sizes
Solving PDEs with DL

• Validation and test accuracy for different model sizes
Solving PDEs with DL

• Validation and test accuracy for different model sizes
Solving PDEs with DL

- Validation and test accuracy for different model sizes
Solving PDEs with DL

- Validation and test accuracy for different model sizes
Additional Examples
Additional Examples

• **Elasticity:** material models
Additional Examples

• **Elasticity**: material models
• **Fluids**: up-res algorithm & dimensionality reduction
Additional Examples

- **Elasticity**: material models
- **Fluids**: up-res algorithm & dimensionality reduction
- By no means exhaustive...
Neural Material - Elasticity

- Learn correction of regular FEM simulation for complex materials

NeoHookean Training

GT: NeoHookean, $E = 2e4$  
Nominal: Co-rotational, $E = 3.5e4$

Ground Truth  
Initial  
Result

Eurographics 2019
Neural Material - Elasticity

• Learn correction of regular FEM simulation for complex materials

NeoHookean Training

GT: NeoHookean, $E = 2e4$

Nominal: Co-rotational, $E = 3.5e4$

Ground Truth    Initial    Result
Neural Material - Elasticity

• Learn correction of regular FEM simulation for complex materials
• “Partial” approach
• Numerical simulation with flexible NN for material behavior
Temporal Data
Temporal Data
Temporal Data

$G(x_a)$

Temporal Data

Temporal Data

\[ X_a \quad X_{t-1} \quad X_t \quad X_{t+1} \]

\[ G(X_a) \quad G(X_{t-1}) \quad G(X_t) \quad G(X_{t+1}) \]

\[ y_{t-1} \quad y_t \quad y_{t+1} \]

\[ D_s \quad D_t \]
Temporal Data

\[ G(x_{t-1}) \]

\[ G(x_{t}) \]

\[ G(x_{t+1}) \]

“Loss” for generator

\[ y_{t-1} \]

\[ y_{t} \]

\[ y_{t+1} \]
Temporal Dat

\[ \begin{align*}
X_a & \quad \rightarrow \quad G \\
X_{t-1} & \quad \rightarrow \quad G \\
X_t & \quad \rightarrow \quad G \\
X_{t+1} & \quad \rightarrow \quad G
\end{align*} \]

\[ y_{t-1} \quad y_t \quad y_{t+1} \]
Temporal Data

\[ D_s \]

\[ G(\mathbf{x}_t) \]

\[ \mathbf{y}_t \sim \mathbf{y}_{t-1} \]

Advection encoded in loss for G

Deep Learning for CG & Geometry Processing
Temporal Data
Temporal Data
Design Options

1. sketching

[Wang, Ceylan, Popovic, Mitra, Siggraph Asia, 2018]
Design Options

1. sketching
2. sewing patterns

[Wang, Ceylan, Popovic, Mitra, Siggraph Asia, 2018]
Design Options

1. sketching
2. sewing patterns
3. draped garment

[Wang, Ceylan, Popovic, Mitra, Siggraph Asia, 2018]
Design Options

1. sketching
2. sewing patterns
3. draped garment

= interaction(sewing pattern, material, body shape)

[Wang, Ceylan, Popovic, Mitra, Siggraph Asia, 2018]
Interaction through Simulation
Interaction through Simulation

realistic simulations but NOT interactive
Multimodal Design

- Sketch
- Garment & Body Parameters
- Shared Shape Space
- Draped Garment
Multimodal Design

- Sketch
- Garment & Body Parameters
- Shared Shape Space
- Draped Garment
- Editing
Learning a Latent Space (AutoEncoder)
Learning a Latent Space (AutoEncoder)
Learning a Latent Space (AutoEncoder)

input sketch ($S$) → DenseNet → sketch descriptor ($\tilde{S}$) → PCA coeff. ($M$) → draped garment ($\tilde{M}$)
Learning a Latent Space (AutoEncoder)
Learning a **Shared** Latent Space (3-way AutoEncoder)

- **input sketch (S)**
- **sketch descriptor (S)**
- **DenseNet**
- **PCA coeff. (M) draped garment (\(\tilde{M}\))**
- **body shape parameters (B)**
- **garment parameters (G)**

\(\{k_{\text{stretch}}, k_{\text{blend}}, k_{\text{shear}}\}\)
Learning a **Shared** Latent Space (3-way AutoEncoder)
Learning a **Shared** Latent Space (3-way AutoEncoder)

- Input sketch (S)
- Sketch descriptor (S)
- DenseNet
- Latent space
  - $f_{S2L}$
  - $f_{P2L}$
  - $f_{L2P}$
- PCA coeff. (M)
- Draped garment ($\tilde{M}$)
- Body shape parameters (B)
- Garment parameters (G)

Eurographics 2019
Loss Function Terms
Loss Function Terms

- Input sketch ($\tilde{S}$)
- Sketch descriptor (S)
- DenseNet
- Latent space
- Loss function terms: $f_{S2L}$, $f_{P2L}$, $f_{L2P}$, $f_{L2M}$
- PCA coefficients (M) draped garment ($\hat{M}$)
- Body shape parameters (B)
- Garment parameters (G)
Loss Function Terms

\[ \| M - f_{L2M}(f_{S2L}(S)) \|_2 \]
Loss Function Terms

\[ \| M - f_{L2M}(f_{S2L}(S)) \|_2 \quad \| P - f_{L2P}(f_{S2L}(S)) \|_2 \]
Loss Function Terms

\[ \| M - f_{L2M}(f_{S2L}(S)) \|_2 \]
\[ \| P - f_{L2P}(f_{S2L}(S)) \|_2 \]
\[ \| M - f_{L2M}(f_{P2L}(P)) \|_2 \]
Loss Function Terms

\[ \| M - f_{L2M}(f_{S2L}(S)) \|_2 \, \| P - f_{L2P}(f_{S2L}(S)) \|_2 \, \| M - f_{L2M}(f_{P2L}(P)) \|_2 \, \| P - f_{L2P}(f_{P2L}(P)) \|_2 \]
Loss Function Terms

\[ L(P, M, S) = \|
M - f_{L2M}(f_{S2L}(S))\|_2 + \|
P - f_{L2P}(f_{S2L}(S))\|_2 + \|
M - f_{L2M}(f_{P2L}(P))\|_2 + \|
P - f_{L2P}(f_{P2L}(P))\|_2 \]
Sketch editing:
Recap

• Checklist for solving PDEs with DL:
Recap

• Checklist for solving PDEs with DL:
  ✓ Model? (Typically given)
Recap

• Checklist for solving PDEs with DL:

  ✓ Model? (Typically given)

  ✓ Data? Can enough training data be generated?
Recap

• Checklist for solving PDEs with DL:

✓ Model? (Typically given)

✓ Data? Can enough training data be generated?

✓ Which NN Architecture?
Recap

• Checklist for solving PDEs with DL:
  ✓ Model? (Typically given)
  ✓ Data? Can enough training data be generated?
  ✓ Which NN Architecture?
  ✓ Fine tuning: learning rate, number of layers & features?
Recap

• Checklist for solving PDEs with DL:

✓ Model? (Typically given)

✓ Data? Can enough training data be generated?

✓ Which NN Architecture?

✓ Fine tuning: learning rate, number of layers & features?

✓ Hyper-parameters, activation functions etc.?
Character Animation

• Learn controllers for character rigs
• Powerful and natural
Character Animation

• Learn controllers for character rigs
• Powerful and natural

[A Deep Learning Framework for Character Motion Synthesis and Editing, SIGGRAPH 2016]

[DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning, SIGGRAPH 2017]

[Mode-Adaptive Neural Networks for Quadruped Motion Control, SIGGRAPH 2018]

[DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills, SIGGRAPH 2018]
Result
Result
Phase-functioned Neural Network

[Holden et al., Siggraph, 2017]
What about Other Creatures?
Footfall Patterns
Gating + Motion Update Network

Frame i-1

Motion Features

Joint Positions
Joint Rotations
Joint Velocities
Trajectory Positions
Trajectory Directions
Trajectory Velocities
Target Velocities
Action Variables

Gating Network

Expert Weights

Motion Prediction Network

Motion Update

Frame i

Joint Positions
Joint Rotations
Joint Velocities
Trajectory Positions
Trajectory Directions
Trajectory Velocities
Root Motion

Blending Coefficients

Blending

Character State and Control Variables

Deep Learning for CG & Geometry Processing
Pace -> Canter ->
Walk -> Turn

Result

41
Pace -> Canter ->
Walk -> Turn
Dynamic Motion Control

[Many of the following slides thanks to Michiel van de Panne]
Dynamic Motion Control

[Many of the following slides thanks to Michiel van de Panne]
In principle:

• specify rewards
• “train” using RL algorithm

\[
\max_\theta \mathbb{E}[\sum_{t=0}^{H} R(s_t)|\pi_\theta]
\]

• use the solution
REINFORCEMENT LEARNING

State description $S$, network structure $\pi$, control $a$, reward $r$, environment physics simulation.

- Structure, torque limits, friction, ...
- Noise amplitude, batch size, step-size control, early termination, learning iterations, ...

Environment physics simulation $\theta$.

Simulation $dt$, control $dt$. 

Deep Learning for CG & Geometry Processing
MOTION IMITATION

- DeepLoco: SIGGRAPH 2017
- DeepMimic: SIGGRAPH 2018
DEEPMIMIC: EXAMPLE-GUIDED DEEP REINFORCEMENT LEARNING OF PHYSICS-BASED CHARACTER SKILLS

• Xue Bin Peng, University of California, Berkeley Pieter Abbeel, University of California, Berkeley Sergey Levine, University of California, Berkeley Michiel van de Panne, University of British Columbia

[SIGGRAPH 2018]
$r_t = \omega^I r_t^I + \omega^G r_t^G$
REWARD

\[ r_t = \omega^I r_t^I + \omega^G r_t^G \]

Imitation Objective
REWARD

\[ r_t = \omega^I r^I_t + \omega^G r^G_t \]

Imitation Objective
REWARD

\[ r_t = \omega^I r^I_t + \omega^G r^G_t \]

Imitation Objective
Imitation Objective

$$r_t = \omega^I r^I_t + \omega^G r^G_t$$
\[ r_t = \omega^I r^I_t + \omega^G r^G_t \]

Imitation Objective

Task Objective
STATE + ACTION

State:
• link positions
• link velocities
• terrain heights
STATE + ACTION

State: 197 D
• link positions
• link velocities
STATE + ACTION

State: 197 D
- link positions
- link velocities

Action:
- PD targets
STATE + ACTION

State: 197 D
- link positions
- link velocities

Action: 36 D
- PD targets
WALKING

[DeepLoco: SIGGRAPH 2017]
WALKING

[DeepLoco: SIGGRAPH 2017]
WALKING

[DeepLoco: SIGGRAPH 2017]
Walking on Conveyor Belts
Walking on Conveyor Belts
Walking on Conveyor Belts
DEEPLocco: HIERARCHICAL RL

1000+ D

HLC

2Hz

imitation reward

LLC

30Hz

SIM

[SIGGRAPH 2017]
Skills From Video: Reinforcement learning of physical skills from video

Transactions on Graphics (Proc. ACM SIGGRAPH Asia 2018)

Xue Bin Peng  Angjoo Kanazawa  Jitendra Malik  Pieter Abbeel  Sergey Levine
University of California, Berkeley
Deep Learning for CG & Geometry Processing

Video → Pose Estimation → Poses → Motion Reconstruction → Reference Motion → Motion Imitation (RL) → $\pi$
Pose Estimation

Video: Handspring A

No Augmentation
[Kanazawa et al. 2018]

With Augmentation
(our work)
Pose Estimation

Video: Handspring A

No Augmentation
[Kanazawa et al. 2018]

With Augmentation
(our work)
Pose Estimation

Video: Handspring A

No Augmentation
[Kanazawa et al. 2018]

With Augmentation
(our work)
SKILLS FROM VIDEOS

Video

Simulation
SKILLS FROM VIDEOS

Video

Simulation
Retargeting

and retarget to different environments.
Retargeting

and retarget to different environments.
Retargeting

and retarget to different environments.
Scalable Muscle-Actuated Human Simulation and Control

SIGGRAPH 2019 Conditional Accept, Seoul National University
Scalable Muscle-Actuated Human Simulation and Control

Seunghwan Lee(1), Kyoungmin Lee(2), Moonseok Park(2), and Jehee Lee(1)
Seoul National University(1), Seoul National University Bundang Hospital(2)
Scalable Muscle-Actuated Human Simulation and Control

Seunghwan Lee(1), Kyoungmin Lee(2), Moonseok Park(2), and Jehee Lee(1)
Seoul National University(1), Seoul National University Bundang Hospital(2)
Scalable Muscle-Actuated Human Simulation and Control

Seunghwan Lee(1), Kyoungmin Lee(2), Moonseok Park(2), and Jehee Lee(1)
Seoul National University(1), Seoul National University Bundang Hospital(2)
"World’s largest Rube Goldberg machine lights up Christmas tree"
https://www.youtube.com/watch?v=RB0qfLVCDv8
"World’s largest Rube Goldberg machine lights up Christmas tree"
https://www.youtube.com/watch?v=RB0qfLVCDv8
Learning **Robustness from Simulations**

[Roussel, Cani, Leon, Mitra, Siggraph, 2019]
Learning Robustness from Simulations

[Roussel, Cani, Leon, Mitra, Siggraph, 2019]
Learning Robustness from Simulations

[Roussel, Cani, Leon, Mitra, Siggraph, 2019]
Learning Robustness from Simulations

[Roussel, Cani, Leon, Mitra, Siggraph, 2019]
Modeling High-dimensional Design Spaces
Modeling High-dimensional Design Spaces

$D^+$

$x$

$D^\partial$ $\partial D^+$ $D^-$

$D^\partial$ $\partial D^+?$ $D^-$?

$D^+?$

$x?$
Online Modeling

Initial exploration → Classifier training → Query synthesis → Probability calibration

\[ X \rightarrow f_k^+ \rightarrow f_k^- \rightarrow D_k \]
Online Modeling

Initial exploration → Classifier training → Query synthesis → Probability calibration

$X \rightarrow f_k^+ \rightarrow f_k \rightarrow D_k$

$X \rightarrow f_{k+1}^+ \rightarrow f_{k+1} \rightarrow D_{k+1}$
Online Modeling

Initial exploration \rightarrow \text{Classifier training} \rightarrow \text{Query synthesis} \rightarrow \text{Probability calibration}

\[
\begin{align*}
X & \quad f_k^+ & \quad f_k^- & \quad D_k \\
X & \quad f_{k+1}^+ & \quad f_{k+1}^- & \quad D_{k+1}
\end{align*}
\]

\cdots
Simple Example
Average Robustness Estimates

$d = 4$

$d = 8$

$d = 11$

$d = 16$

Error $\epsilon$ on the output layout parameters $x^*$

CreativeAI: Editable 3D Content Creation
Code Examples

PCA/SVD basis
Linear Regression
Polynomial Regression
Stochastic Gradient Descent vs. Gradient Descent
Multi-layer Perceptron
Edge Filter ‘Network’
Convolutional Network
Filter Visualization
Weight Initialization Strategies
Colorization Network
Autoencoder
Variational Autoencoder
Generative Adversarial Network

http://geometry.cs.ucl.ac.uk/dl_for_CG/
## Timetable

<table>
<thead>
<tr>
<th></th>
<th>Niloy</th>
<th>Federico</th>
<th>Iasonas</th>
<th>Emanuele</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>9:00</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td><strong>Machine Learning Basics</strong></td>
<td>~9:05</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Neural Network Basics</strong></td>
<td>~9:35</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td><strong>Alternatives to Direct Supervision (GANs)</strong></td>
<td>~11:00</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td><strong>Image Domain</strong></td>
<td>~11:45</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td><strong>3D Domains (extrinsic)</strong></td>
<td>~13:30</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3D Domains (intrinsic)</strong></td>
<td>~14:15</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td><strong>Physics and Animation</strong></td>
<td>~16:00</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Discussion</strong></td>
<td>~16:45</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Sessions: A. 9:00-10:30 (coffee) B. 11:00-12:30 [LUNCH] C. 13:30-15:00 (coffee) D. 15:30-17:00
Course Information (slides/code/comments)

http://geometry.cs.ucl.ac.uk/dl_for_CG/