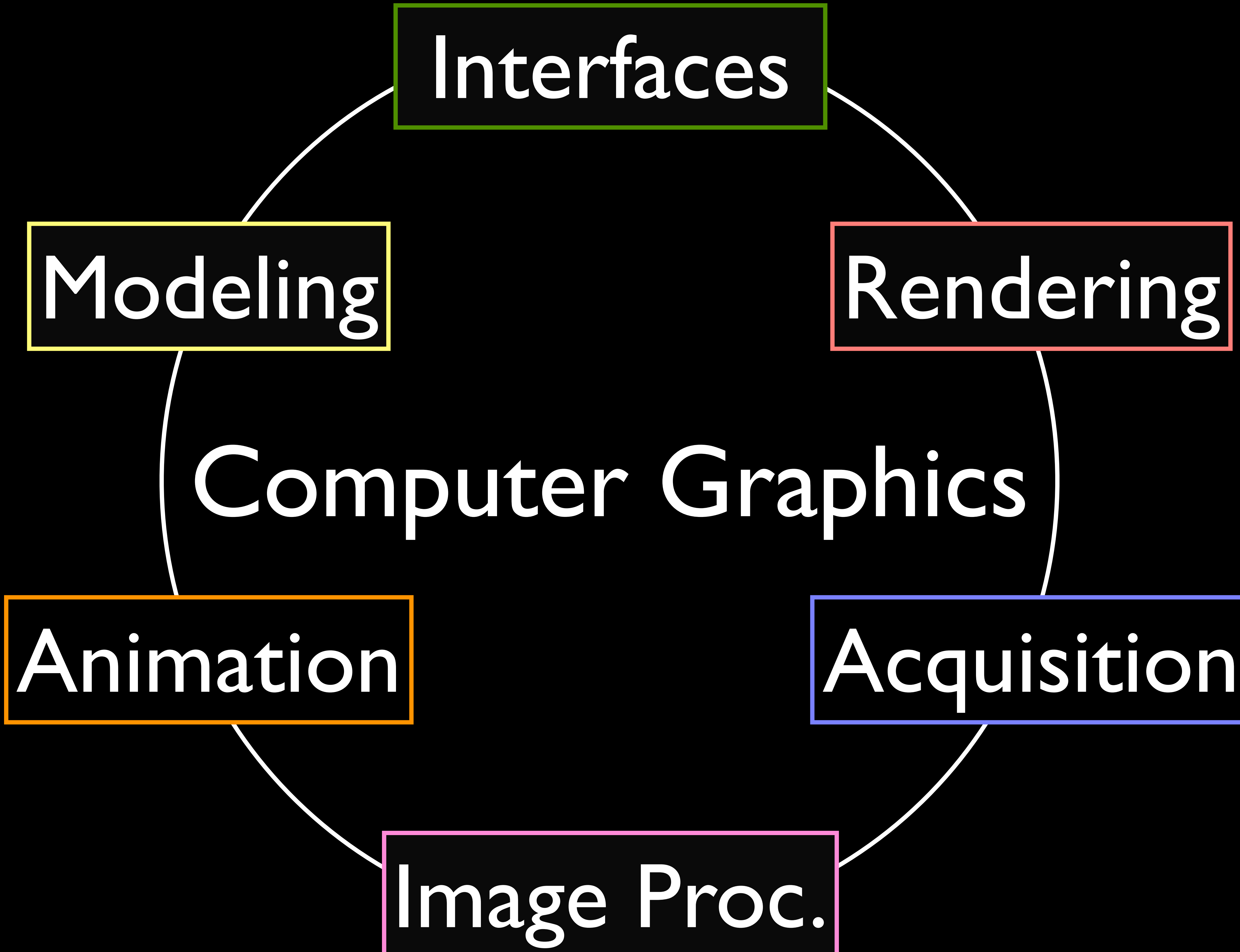
# Thinking Outside the Cornell Box

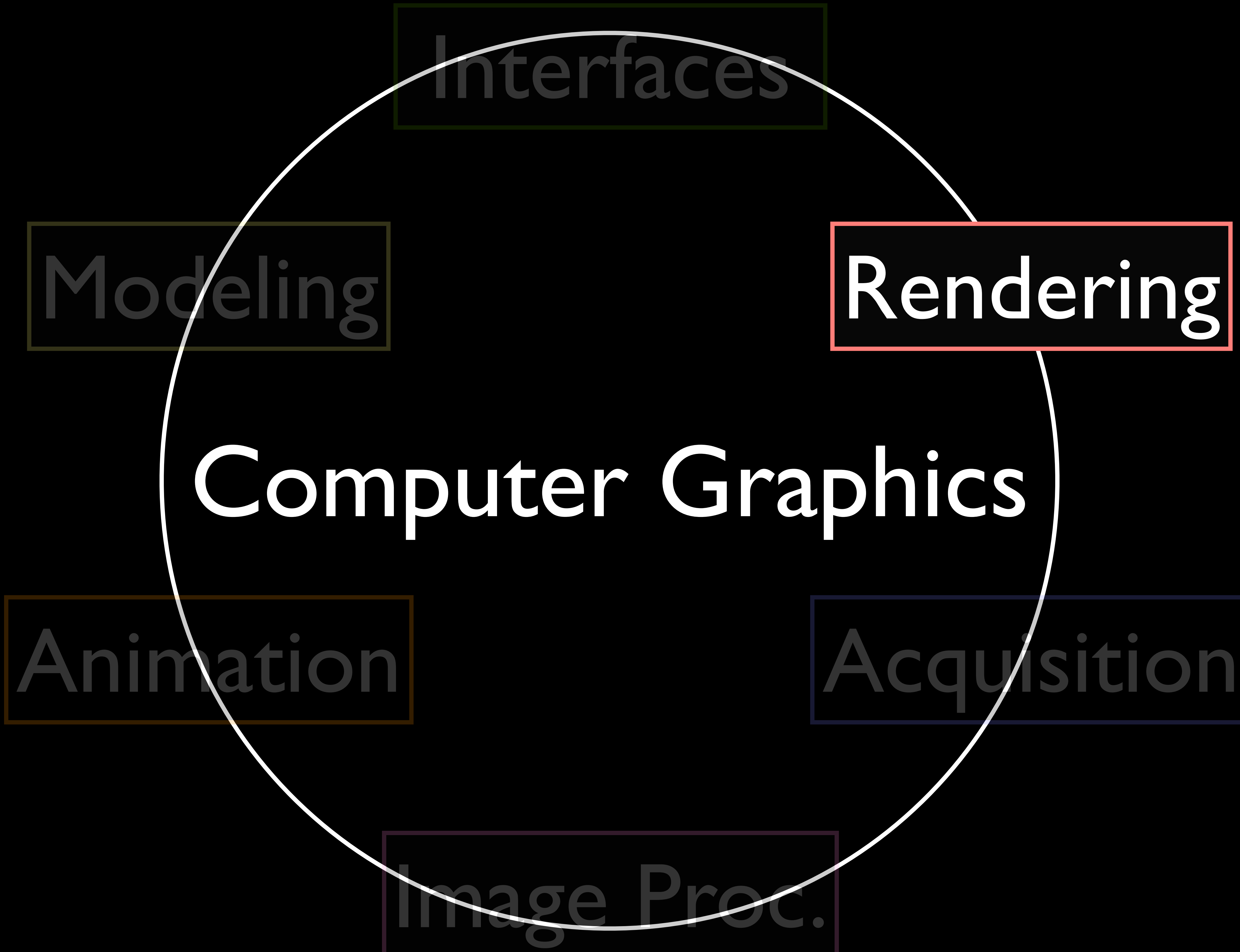
## Non-rendering Research by a Rendering Guy

**Toshiya Hachisuka**

**University of Tokyo**

**MEIS 2017**


Interfaces
Modeling
Rendering
Computer Graphics
Animation
Acquisition
Image Proc.


Interfaces
Modeling
Rendering
Computer Graphics
Animation
Acquisition
Image Proc.

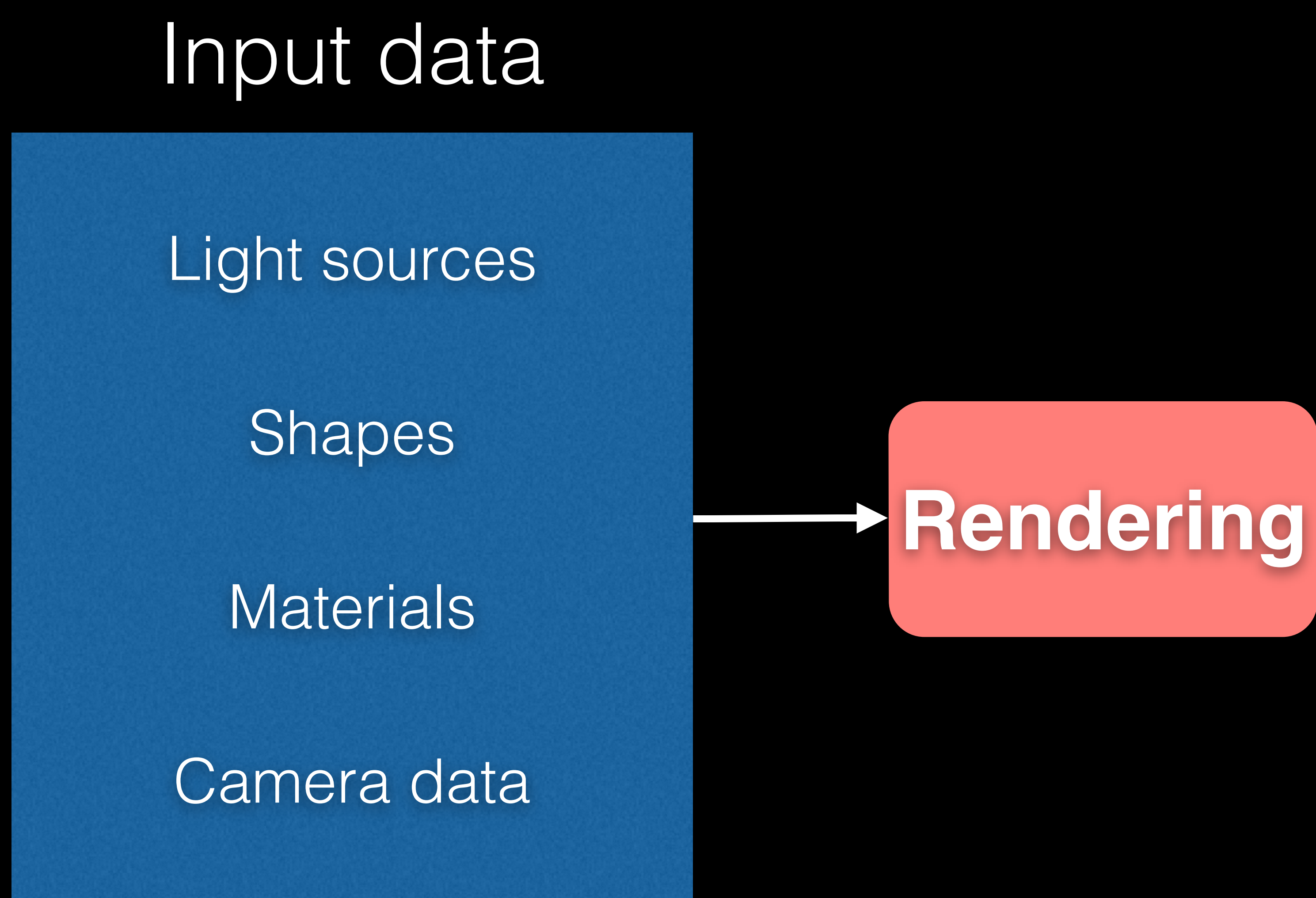
## Input data

Light sources

Shapes

Materials

Camera data


Input data
Light sources
Shapes
Materials
Camera data
Rendering

Input data

Light sources

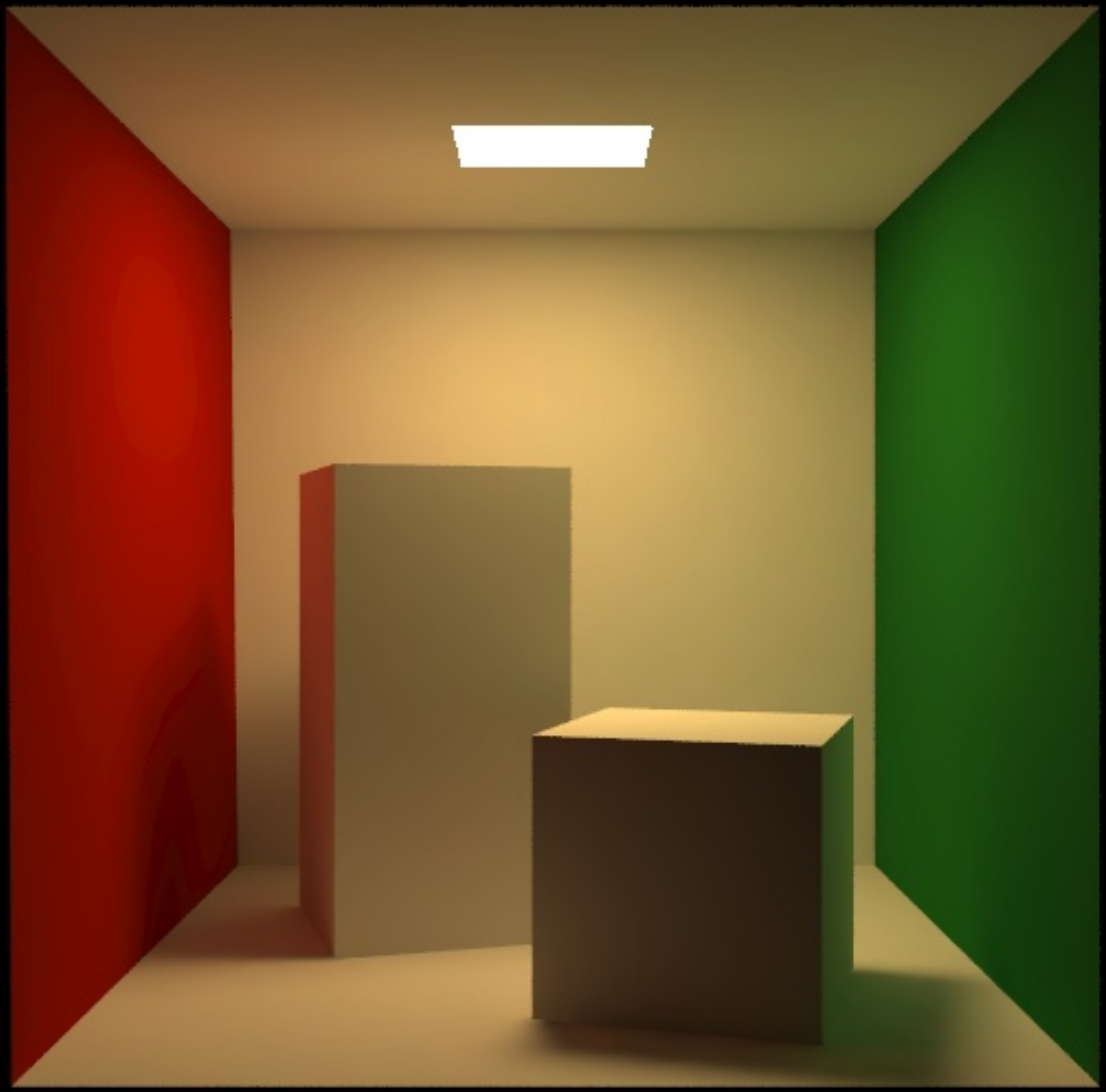
Shapes

Materials

Camera data

→ **Rendering** →

Image



Cornell box

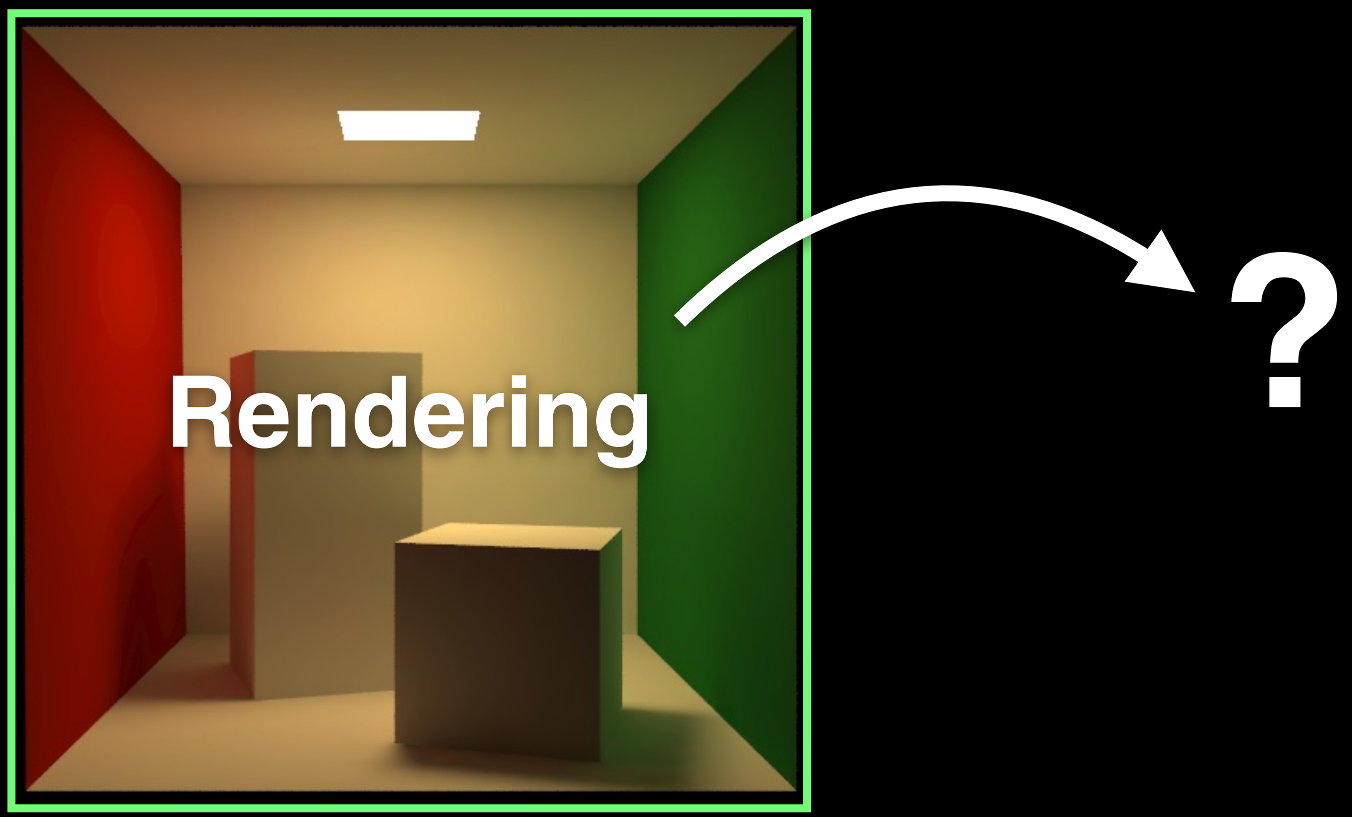

12
3
6

# Rendering guy

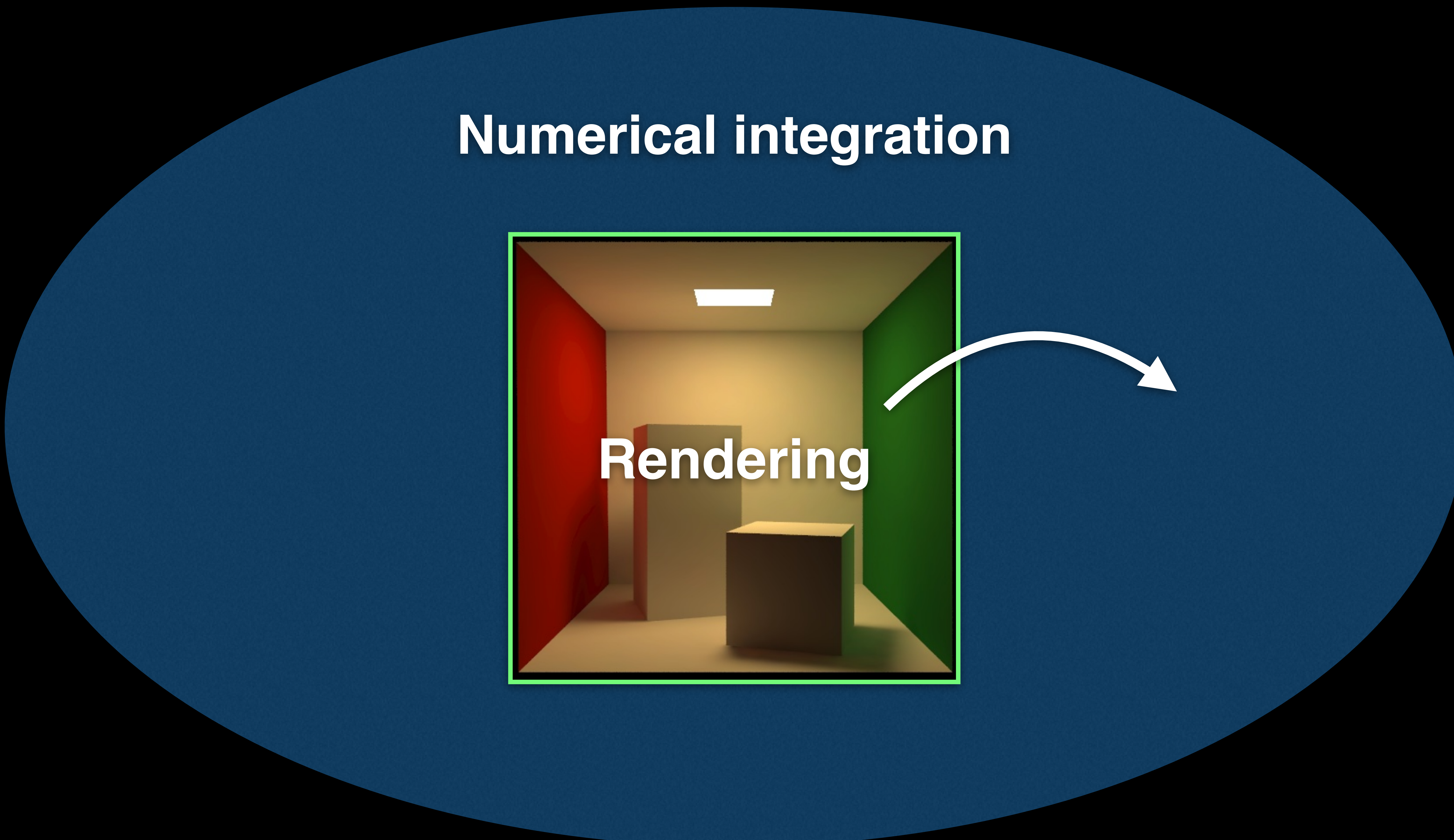
- I have been working on various topics in rendering
  - Over the past ten years
  - ~95% of my publications are on rendering
  - People probably see me as “a rendering guy”

# Thinking inside the box

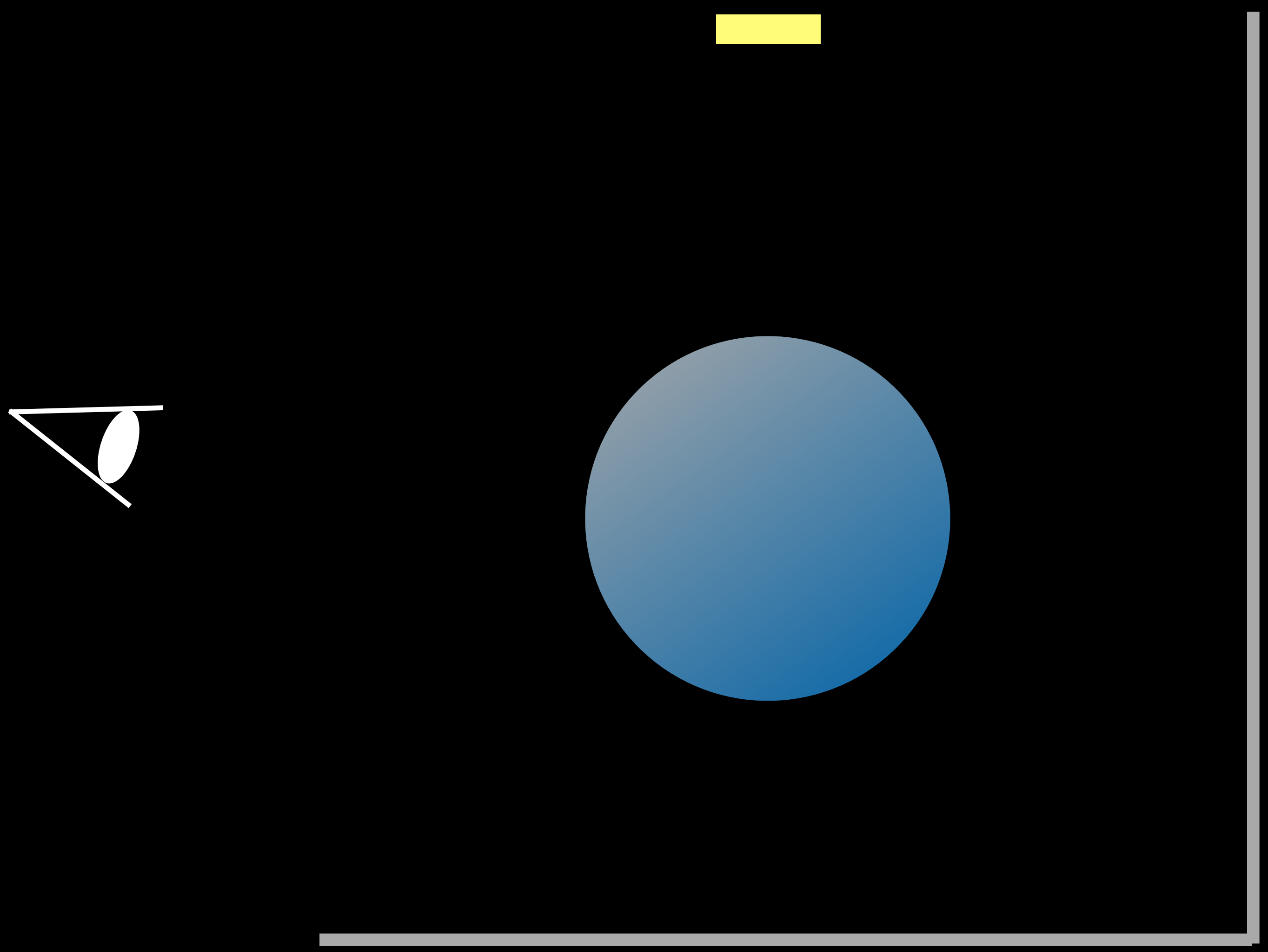
# Thinking outside the box



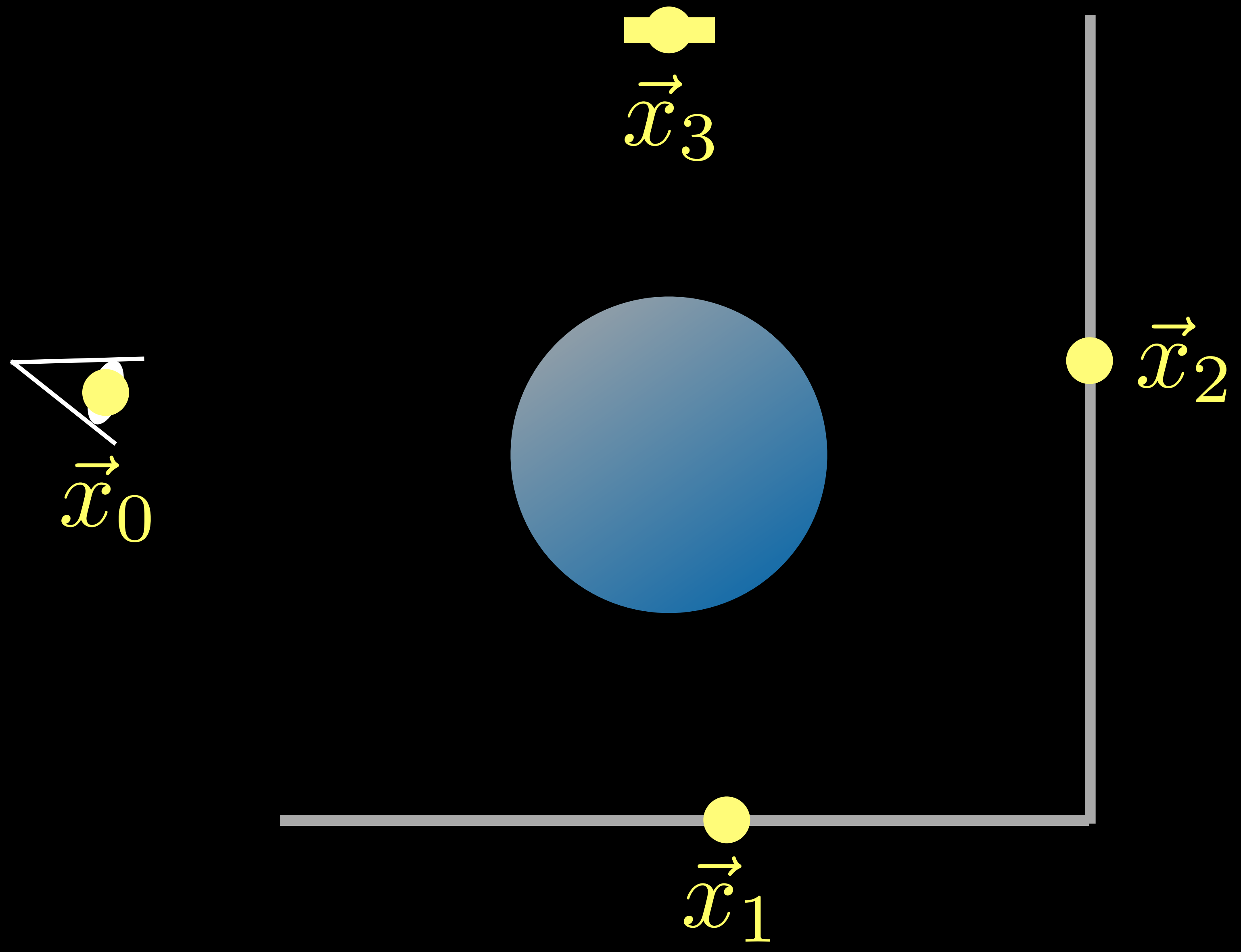
# Thinking outside the box



# Rendering as numerical integration

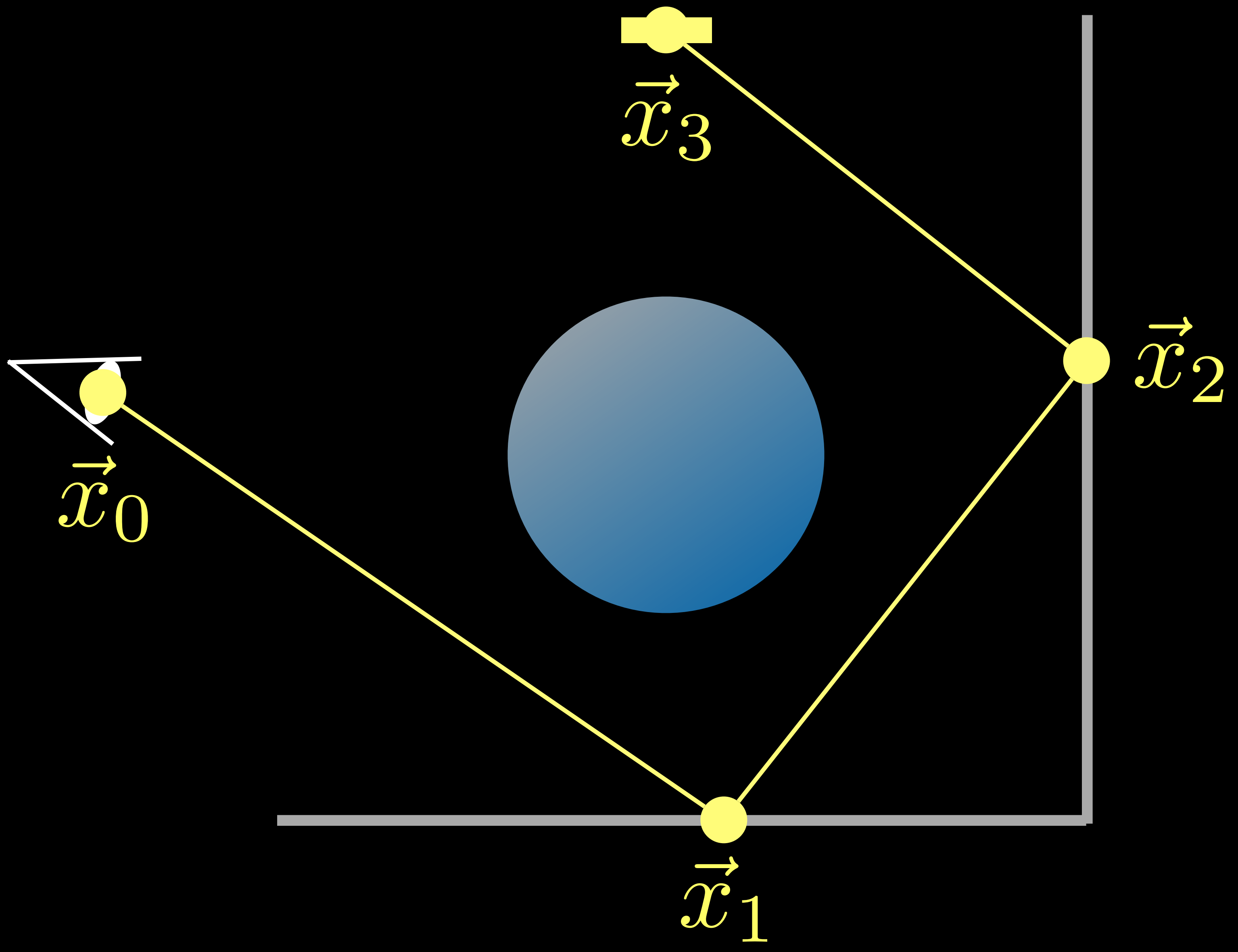


# Rendering as numerical integration



# Rendering as numerical integration

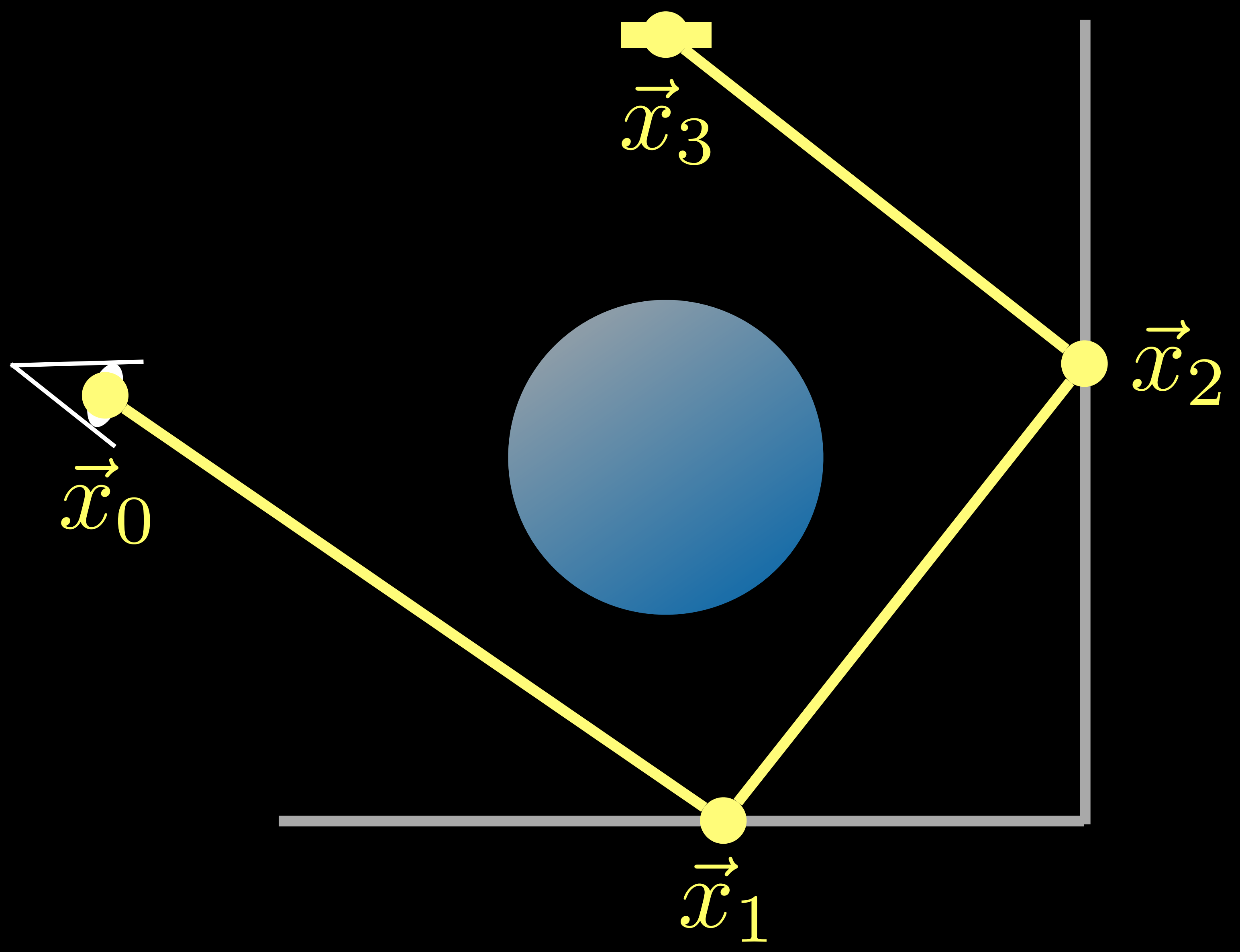
- Represent a path as a vector




$$(\vec{x}_0, \vec{x}_1, \vec{x}_2, \vec{x}_3)$$

# Rendering as numerical integration

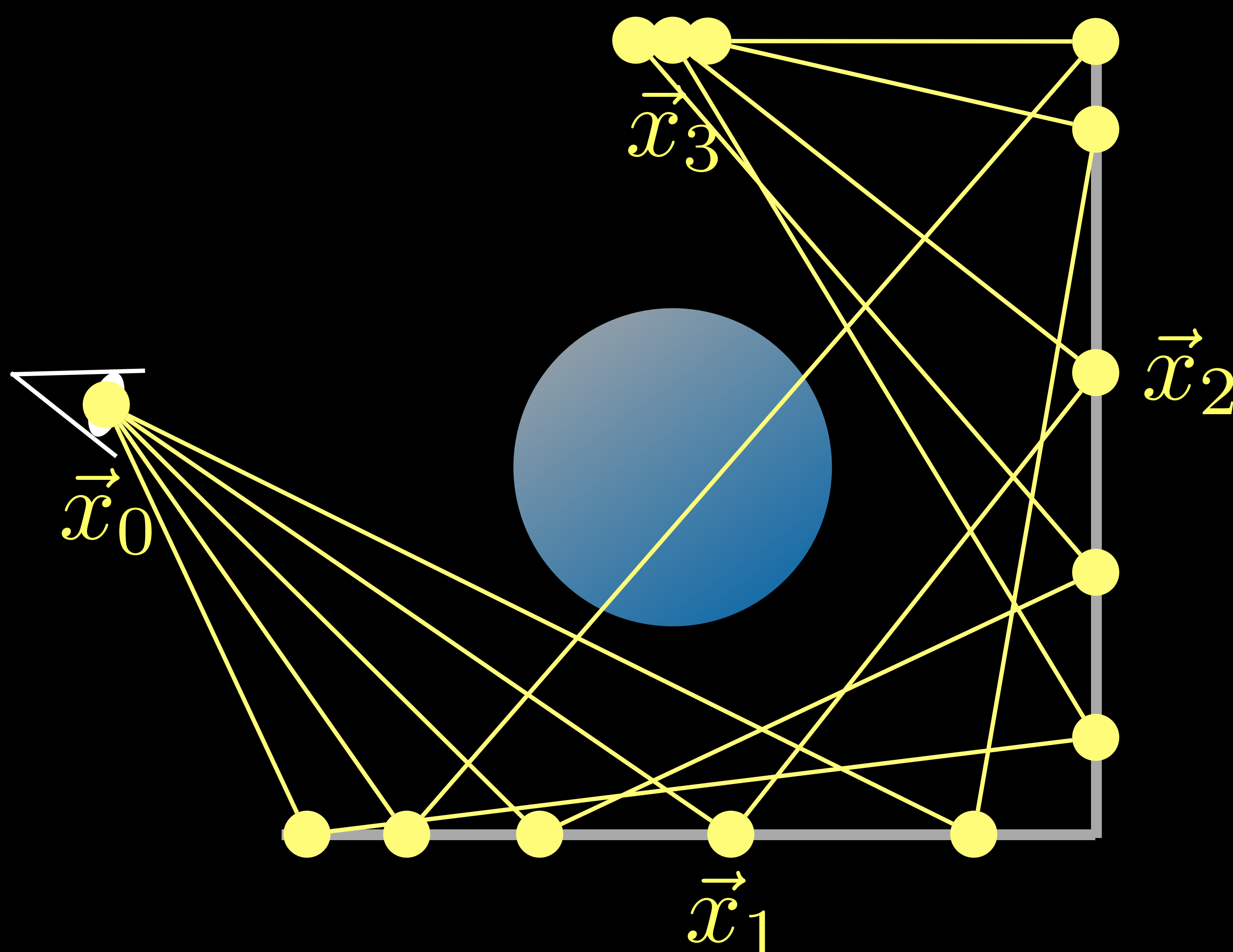
- Define how much light is carried along the path




$$T(\vec{x}_0, \vec{x}_1, \vec{x}_2, \vec{x}_3)$$

# Rendering as numerical integration

- Total amount of light = integration over paths




$$\int T(\vec{x}_0, \vec{x}_1, \vec{x}_2, \vec{x}_3)\, dA$$

# Rendering as numerical integration

- Solve numerically by taking N samples
  - Numerical integration problem

$$\sum_{j=1}^{N} T(\vec{x}_{0_j}, \vec{x}_{1_j}, \vec{x}_{2_j}, \vec{x}_{3_j}) dA_j \approx \int T(\vec{x}_0, \vec{x}_1, \vec{x}_2, \vec{x}_3) dA$$

# Thinking outside the box

- My works can be seen as general numerical methods
  - Progressive density estimation [2008, 2009]
  - Markov chain Monte Carlo methods [2011, 2014]
  - Numerical integration methods [2013, 2016]

# Thinking outside the box

- My works can be seen as general numerical methods
  - Progressive density estimation [2008, 2009]
  - Markov chain Monte Carlo methods [2011, 2014]
  - Numerical integration methods [2013, 2016]

Rendering methods applicable to problems outside rendering

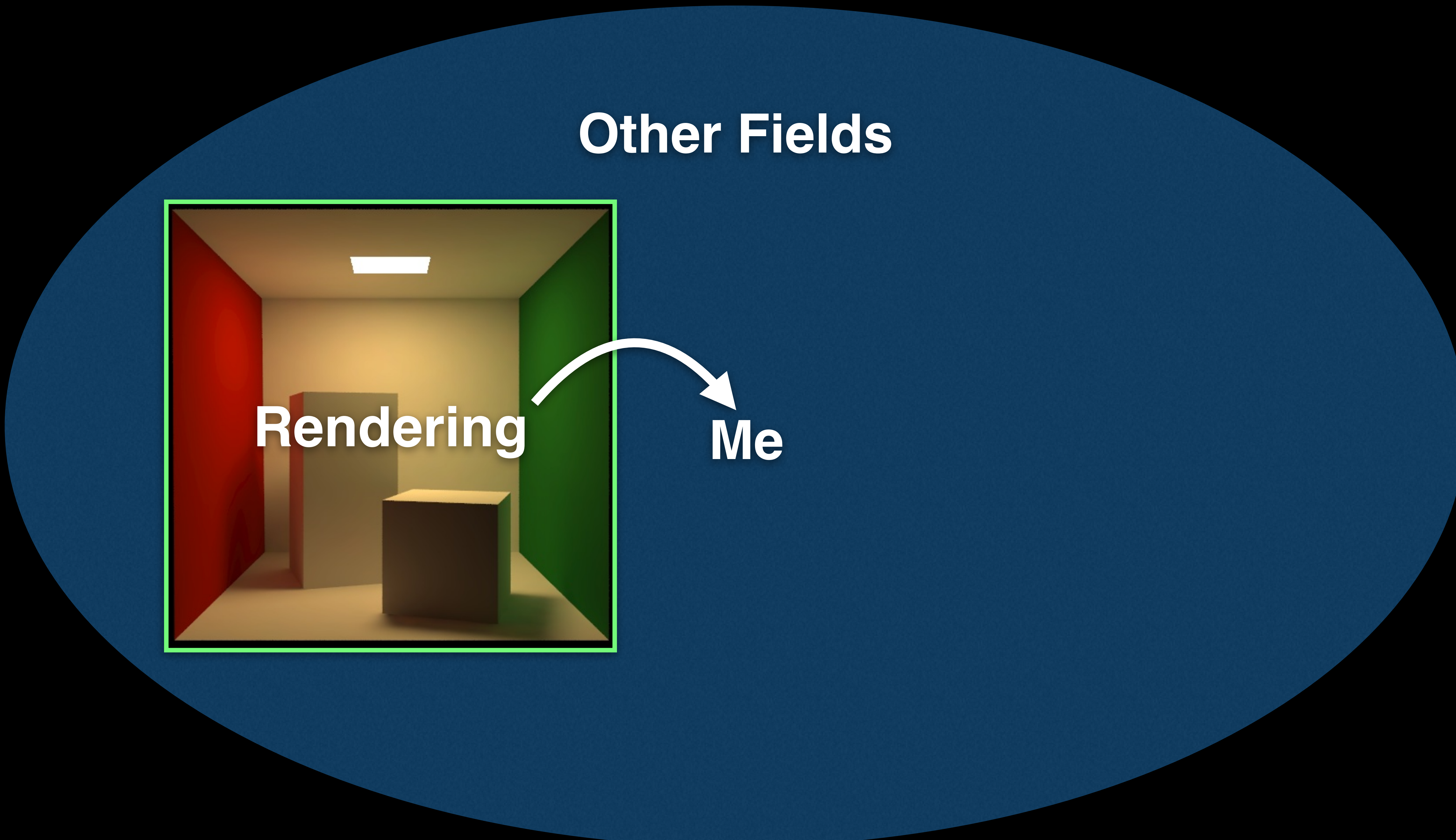
# Thinking outside the box

- My works can be seen as general numerical methods
  - Progressive density estimation [2008, 2009]
  - Markov chain Monte Carlo methods [2011, 2014]
  - Numerical integration methods [2013, 2016]

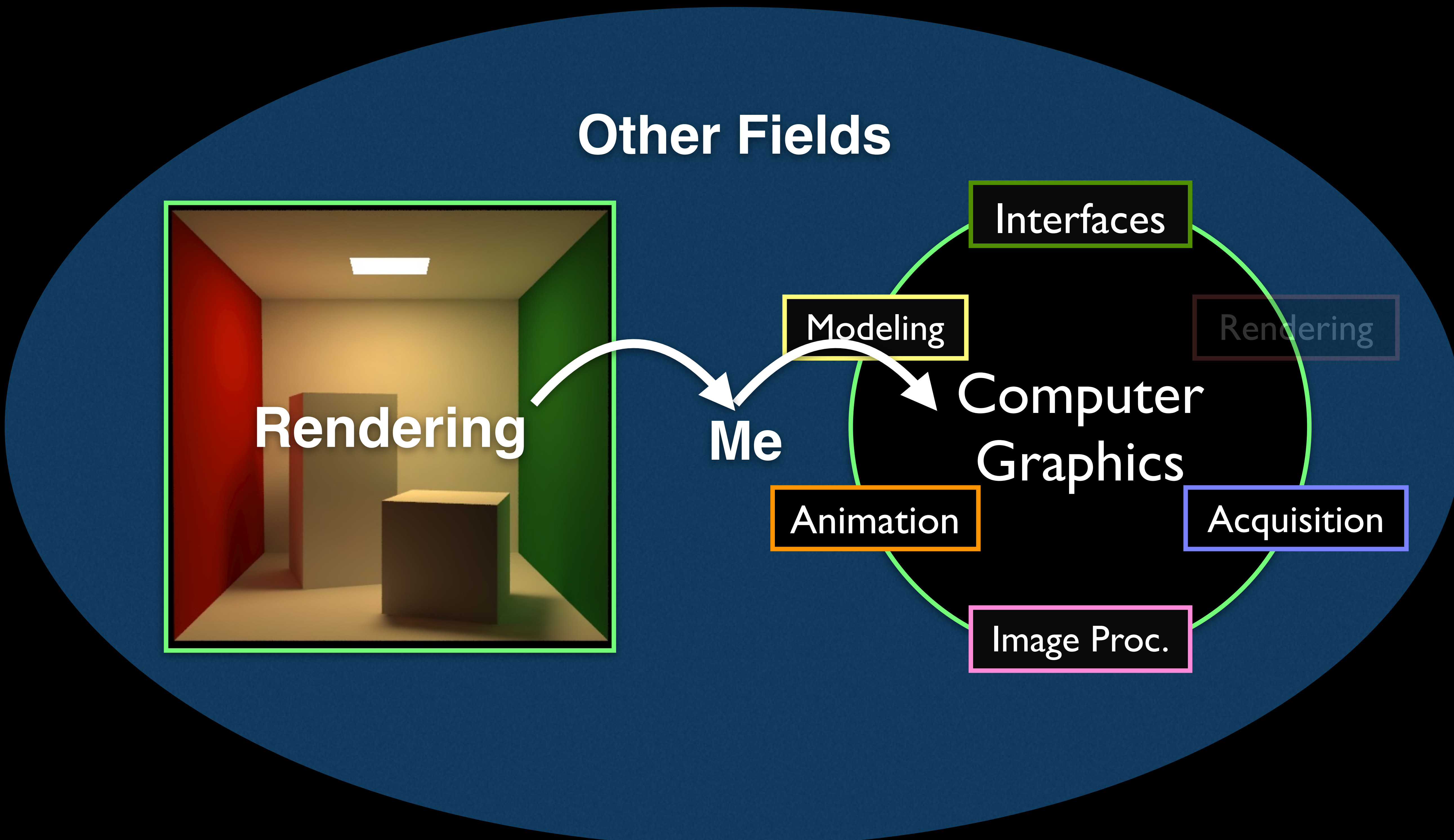
Rendering methods applicable to problems outside rendering

**and then stepping back even more…**

# Thinking outside the box



# Thinking outside the box



# Thinking outside the box

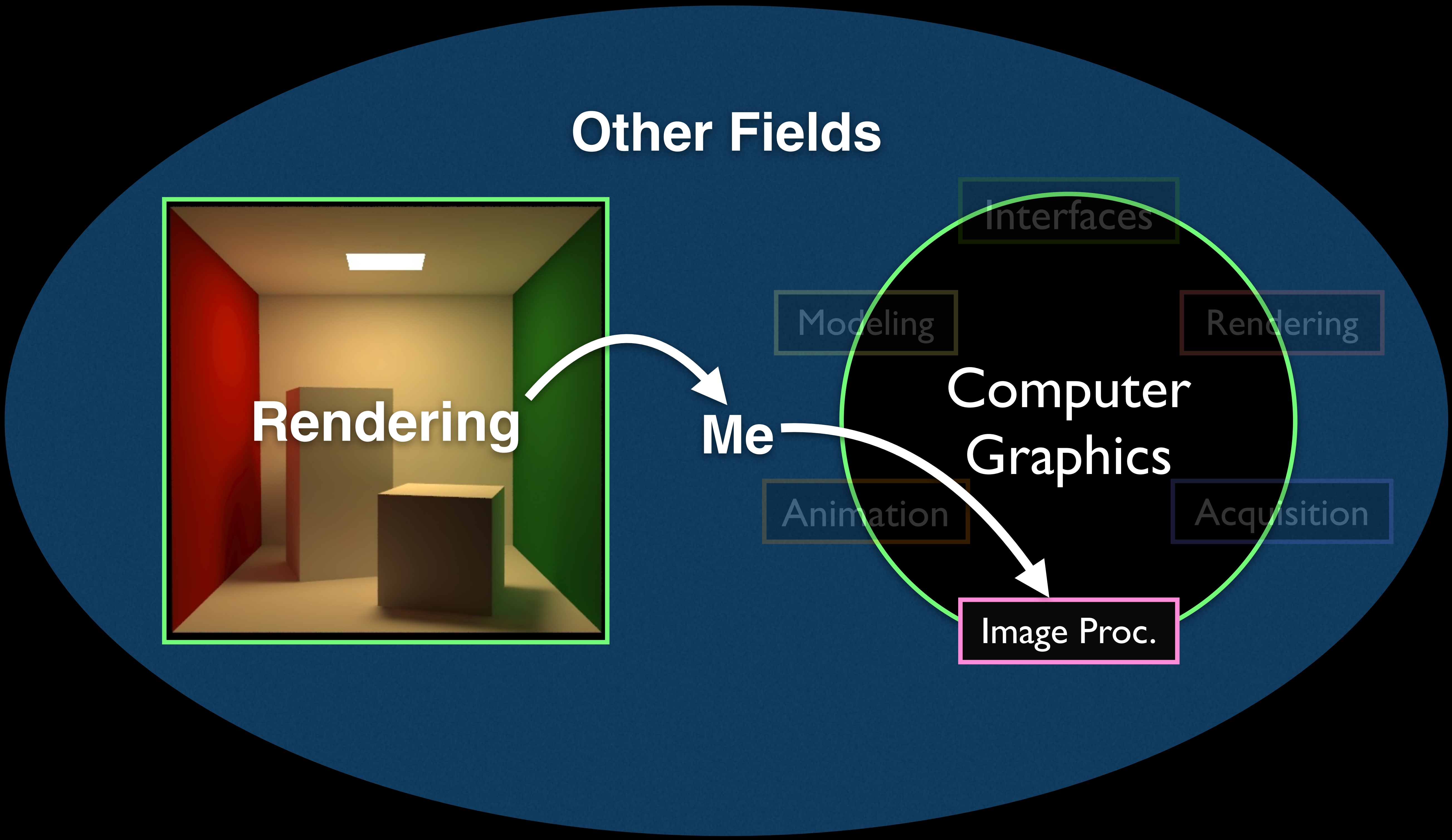


Image Proc.

# “Wavelet Convolutional Neural Networks”

S. Fujieda, K. Takayama, and T. Hachisuka
arXiv:1707.07394, July 2017

# Idea

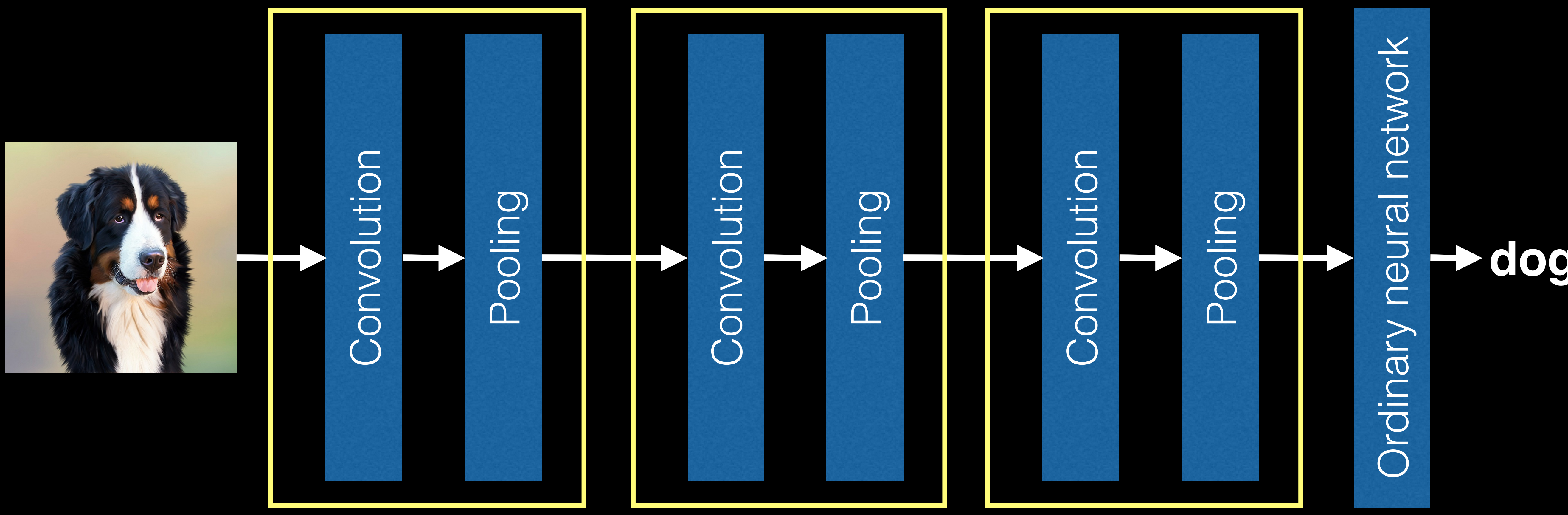
Image processing: Wavelet analysis of images

+

Machine learning: Convolutional neural networks

# Convolutional Neural Networks

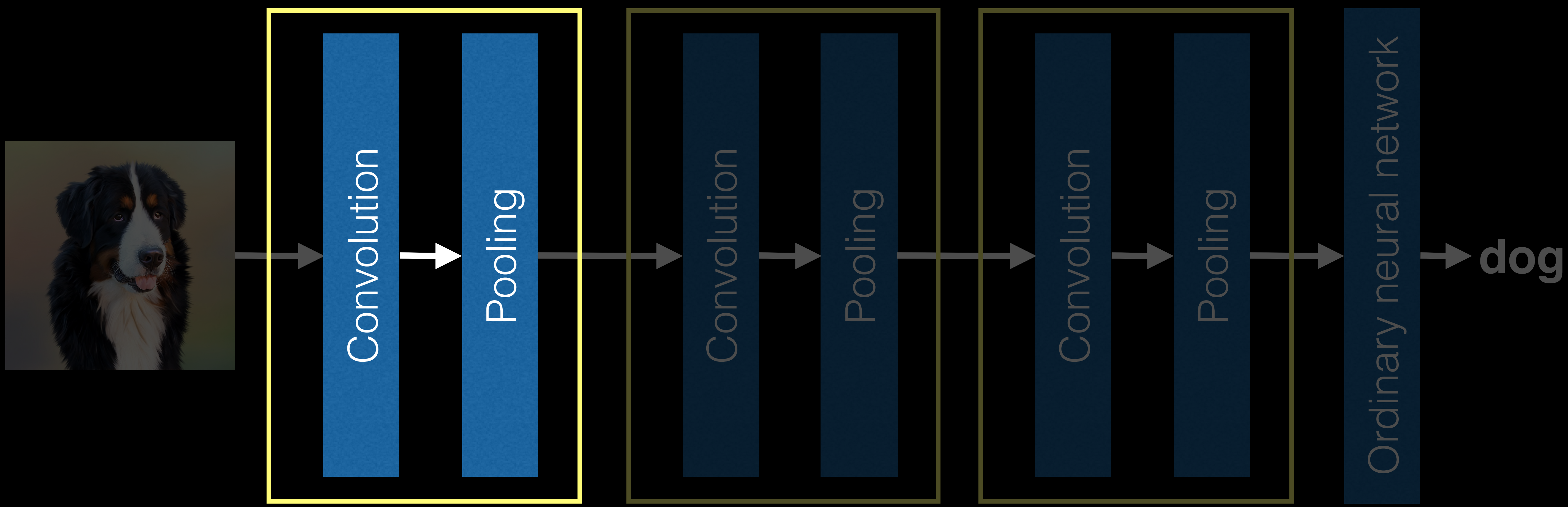
- Most popular network architecture for images




"ImageNet Classification with Deep Convolutional Neural Networks", Krizhevsky et al., NIPS 2012

# Convolutional Neural Networks

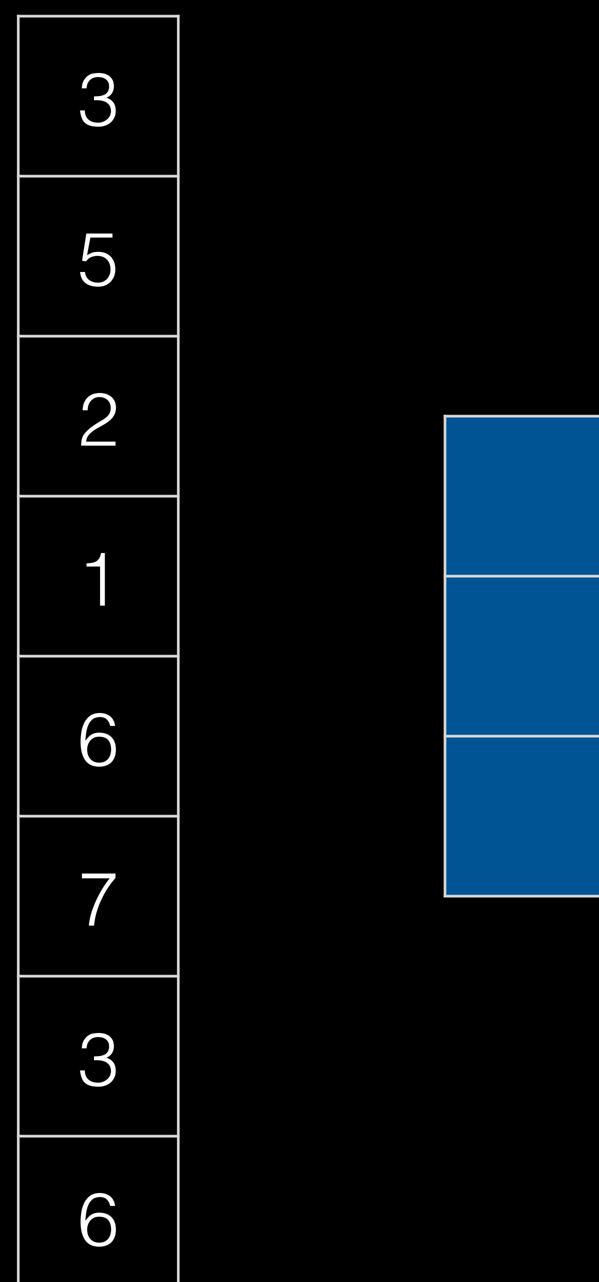
- Most popular network architecture for images




“ImageNet Classification with Deep Convolutional Neural Networks”, Krizhevsky et al., NIPS 2012

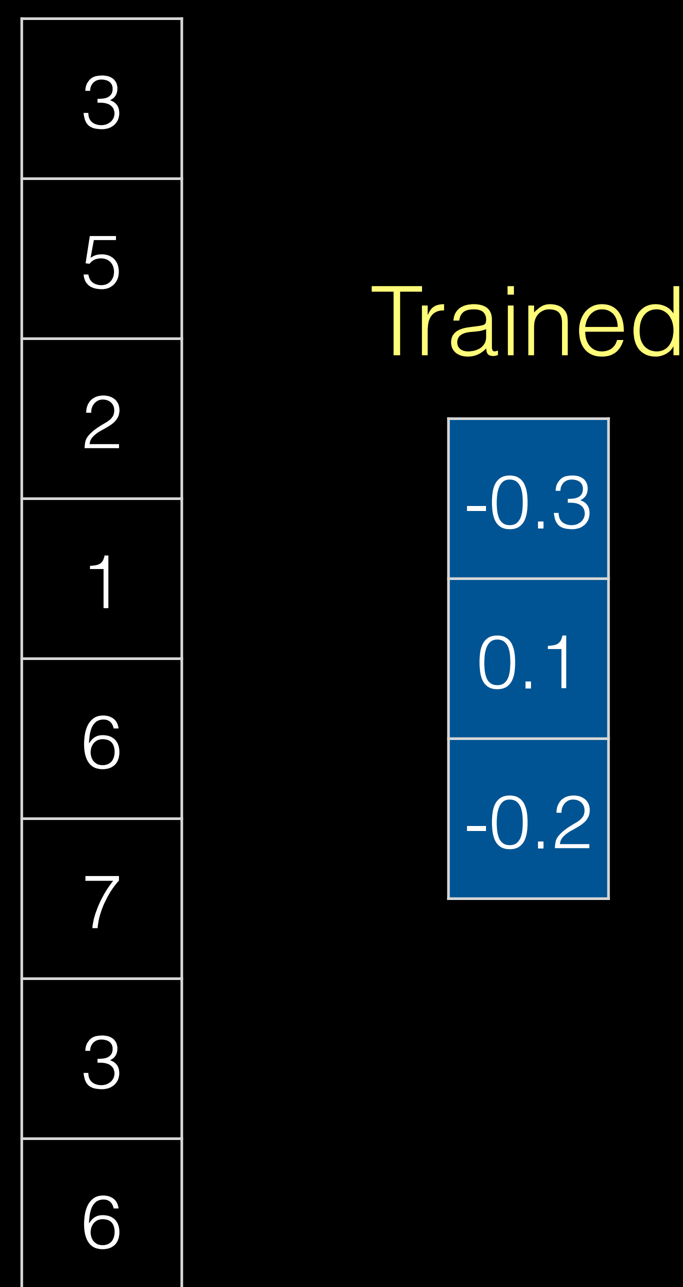
# Convolution

- Filter the input by a shared, trainable kernel



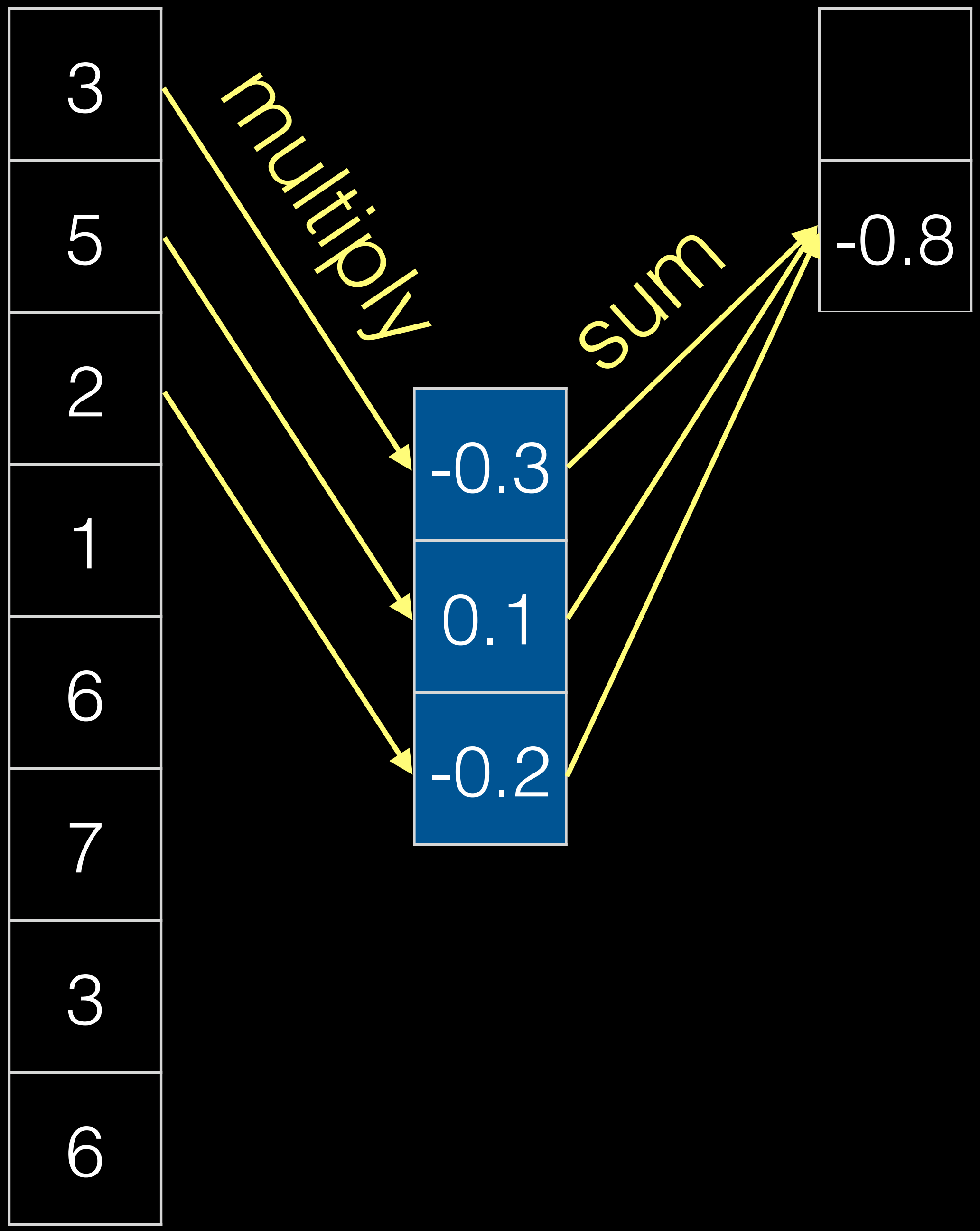
# Convolution

- Filter the input by a shared, trainable kernel



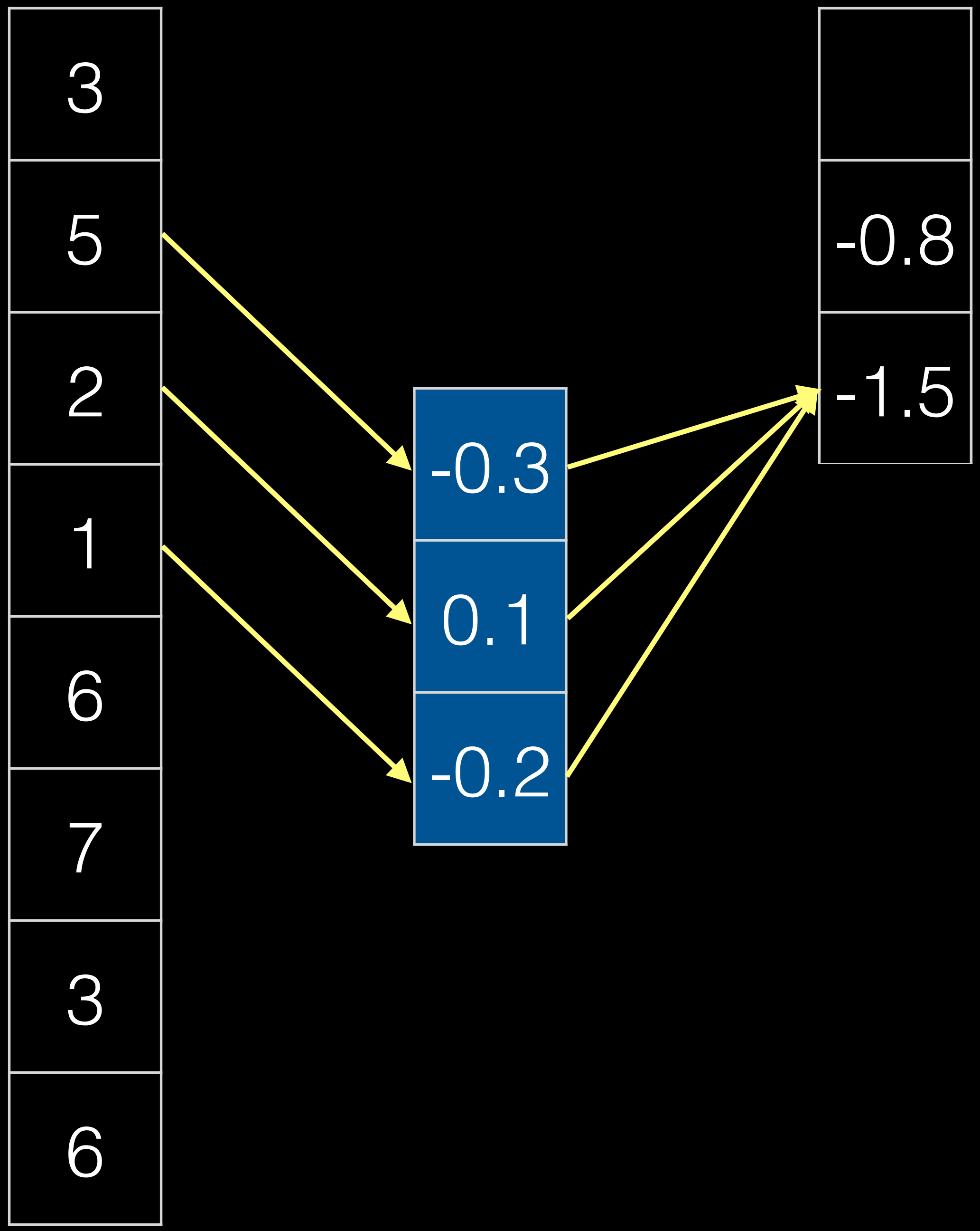
# Convolution

- Filter the input by a shared, trainable kernel



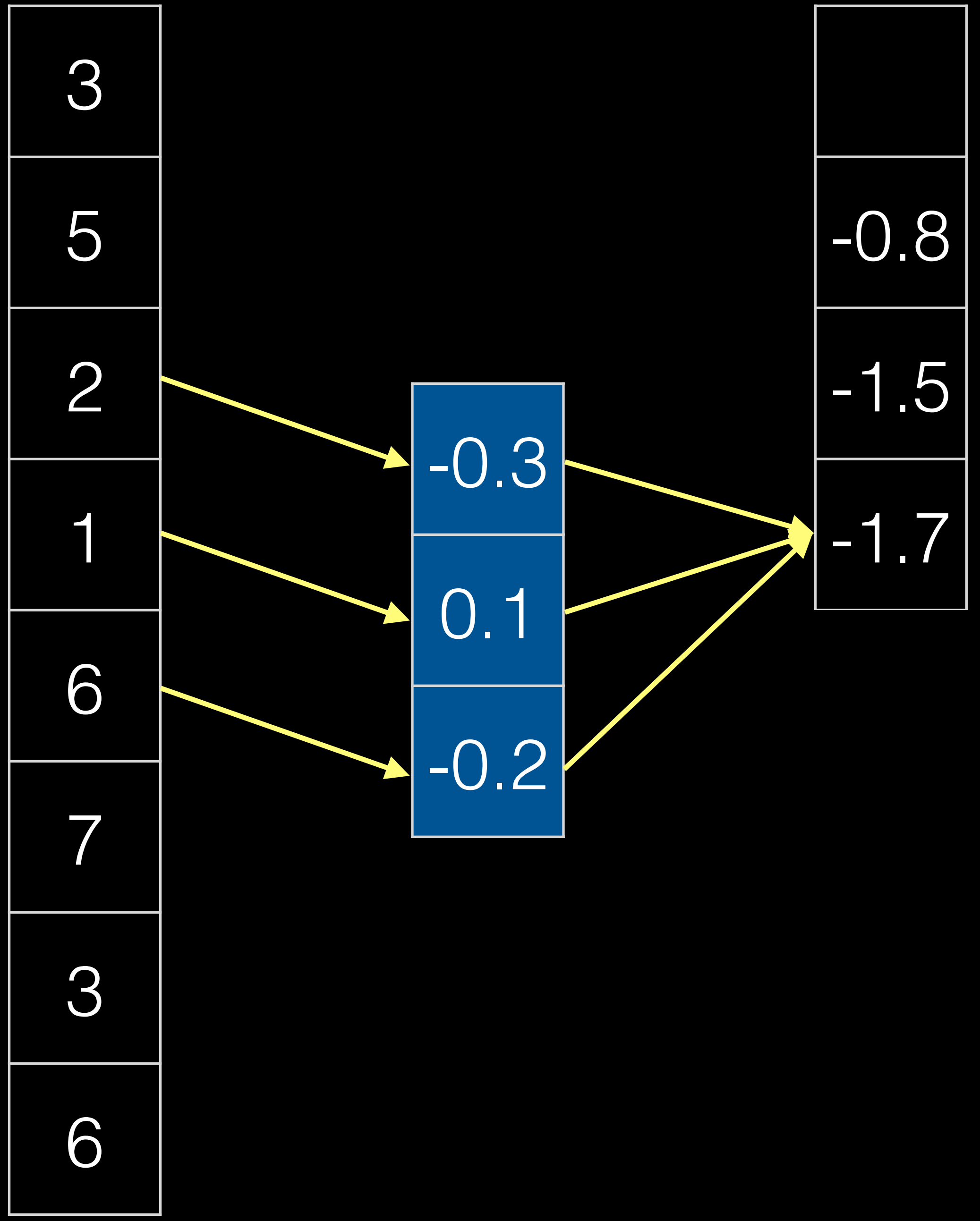
# Convolution

- Filter the input by a shared, trainable kernel



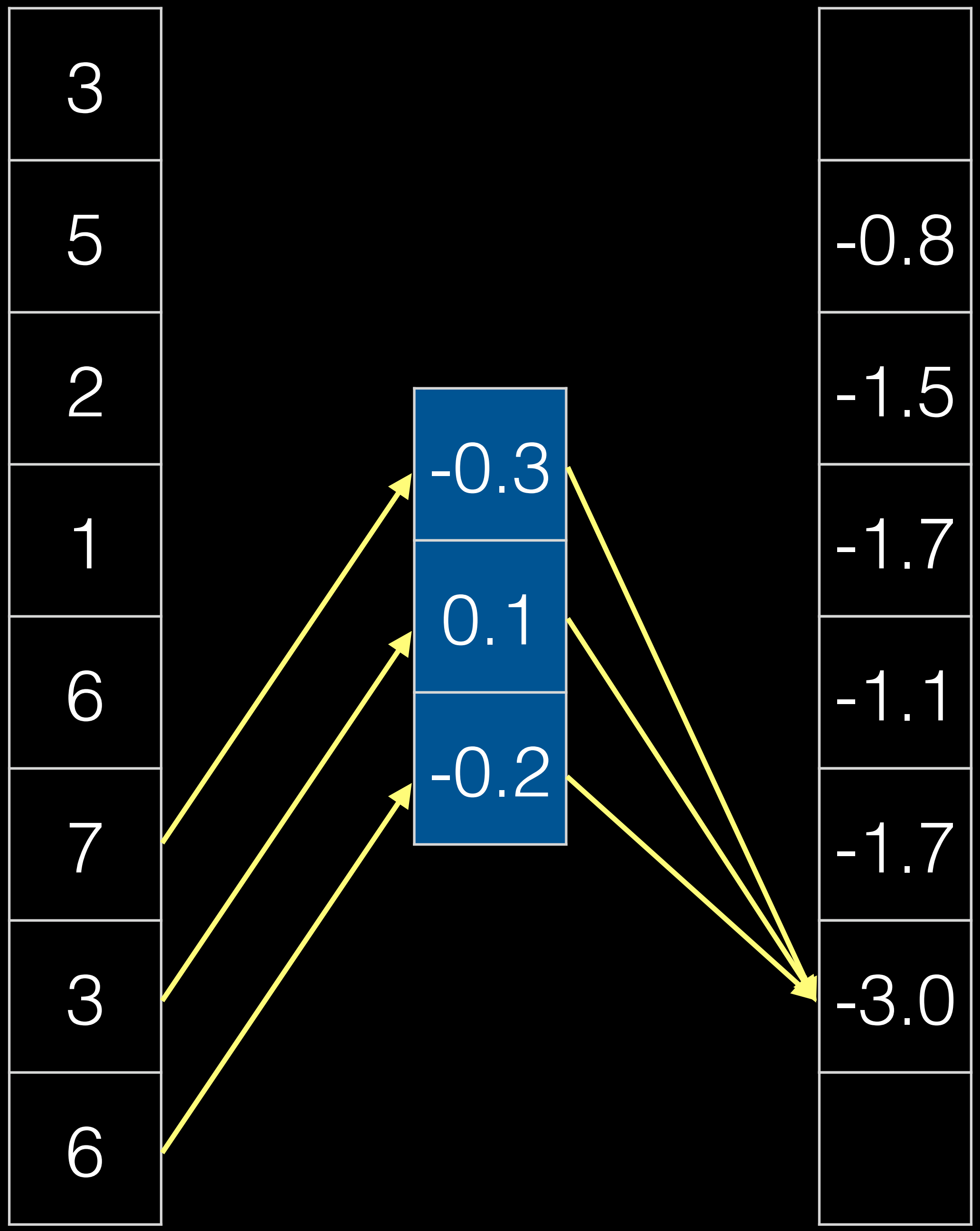
# Convolution

- Filter the input by a shared, trainable kernel



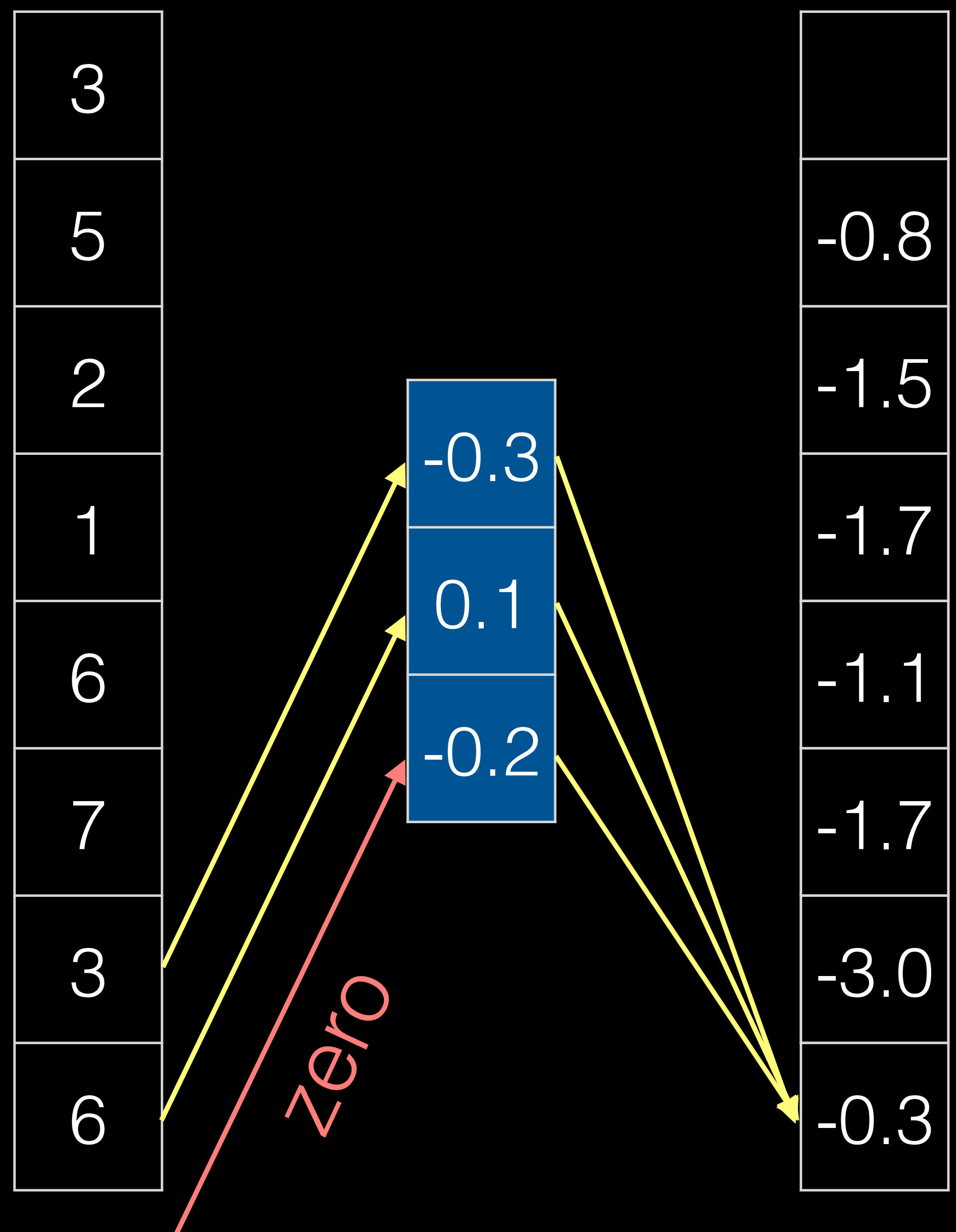
# Convolution

- Filter the input by a shared, trainable kernel



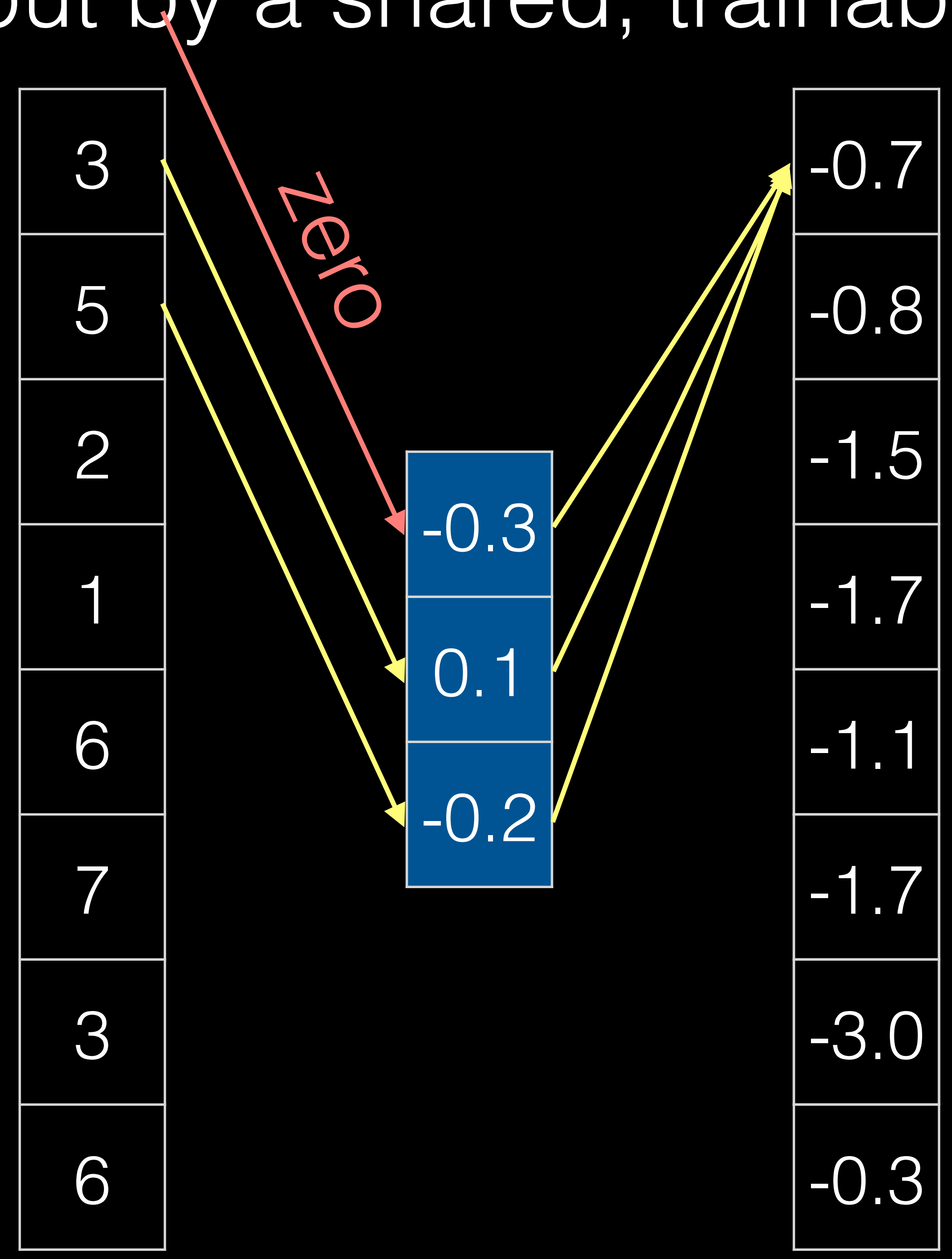
# Convolution

- Filter the input by a shared, trainable kernel



# Convolution

- Filter the input by a shared, trainable kernel



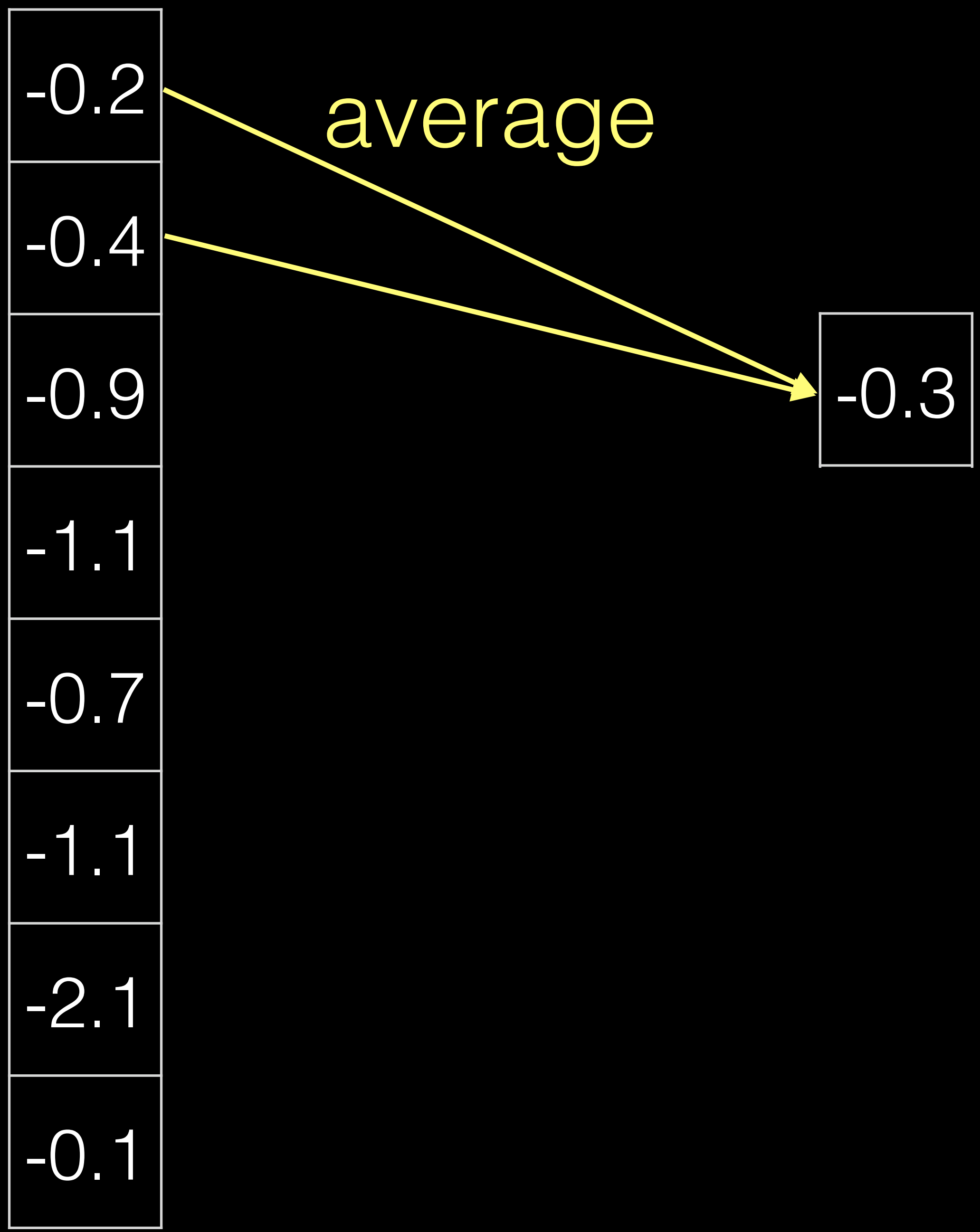
# Pooling

- Fixed aggregating operation of the input

| |
|---|
| -0.2 |
| -0.4 |
| -0.9 |
| -1.1 |
| -0.7 |
| -1.1 |
| -2.1 |
| -0.1 |

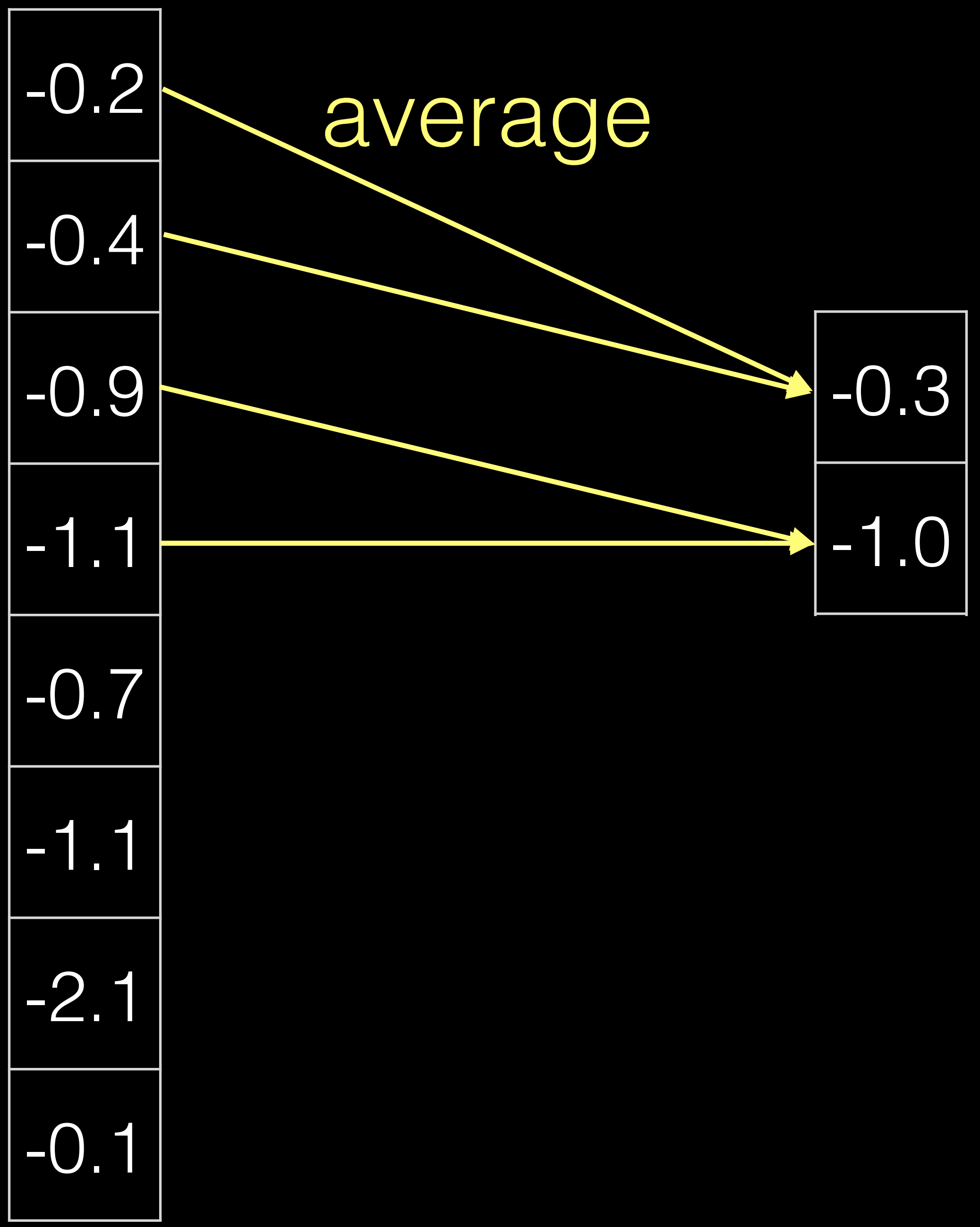
# Pooling

- Fixed aggregating operation of the input



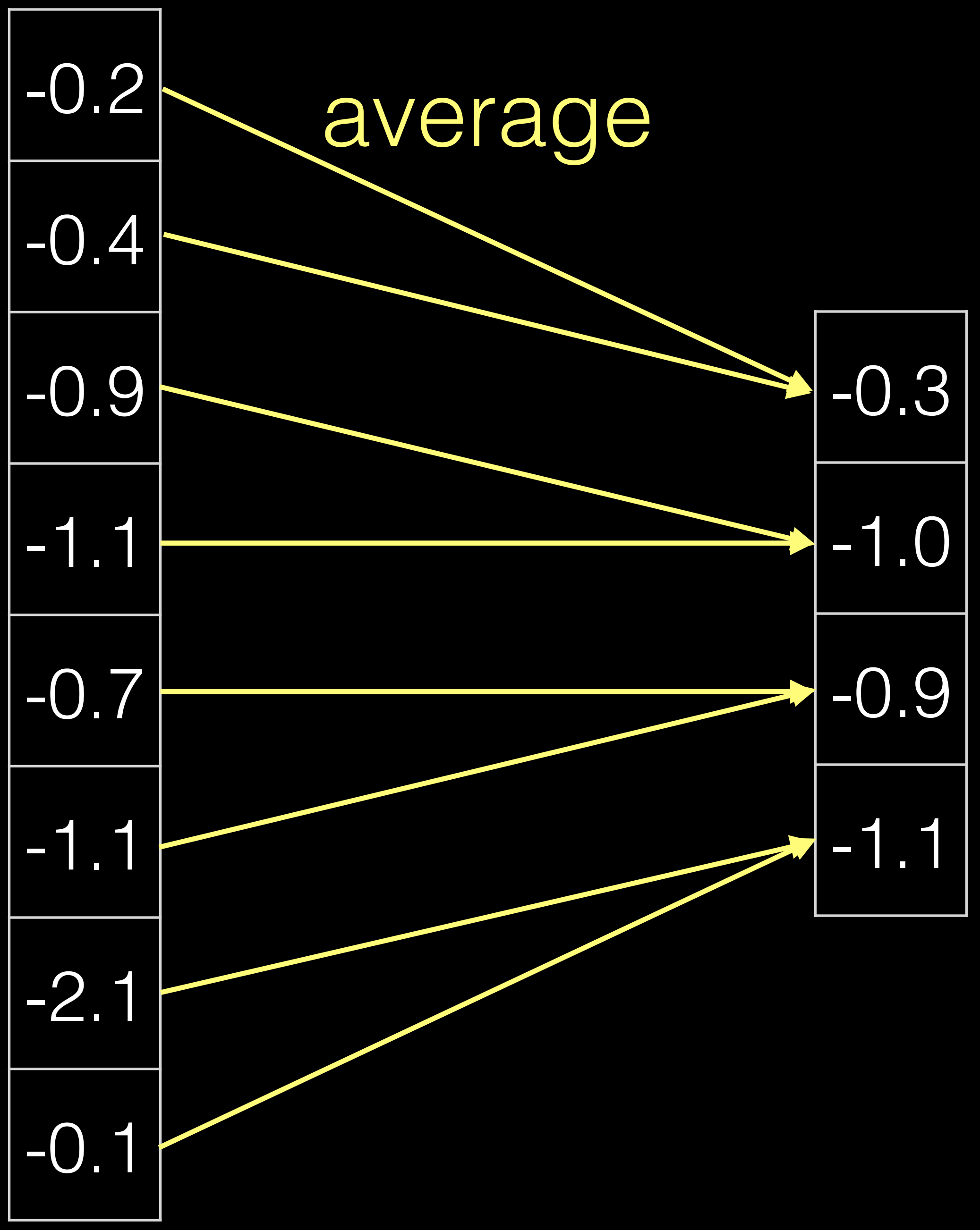
# Pooling

- Fixed aggregating operation of the input



# Pooling

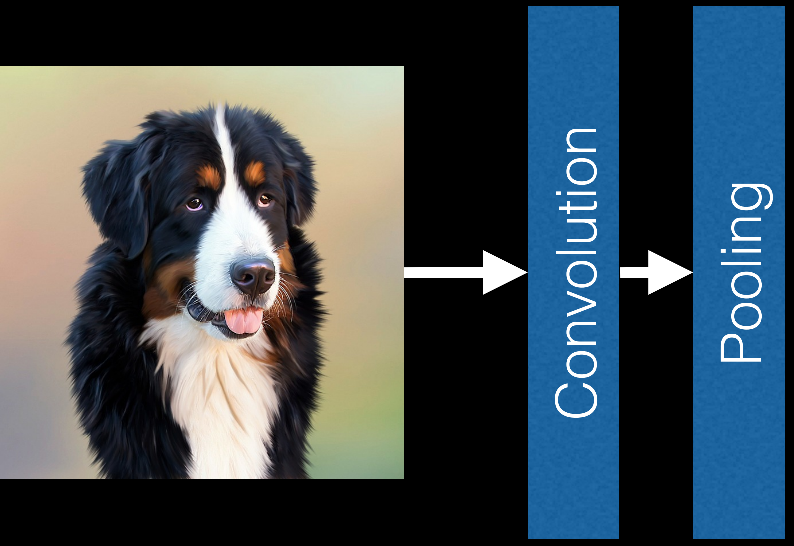
- Fixed aggregating operation of the input

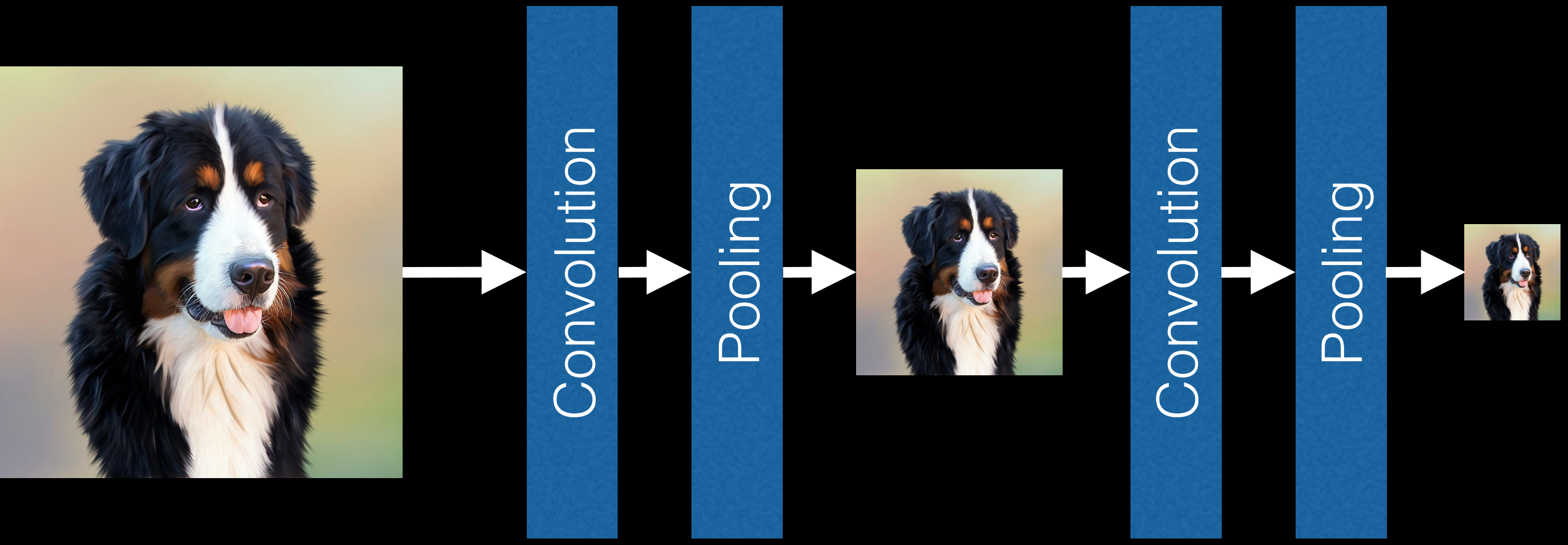


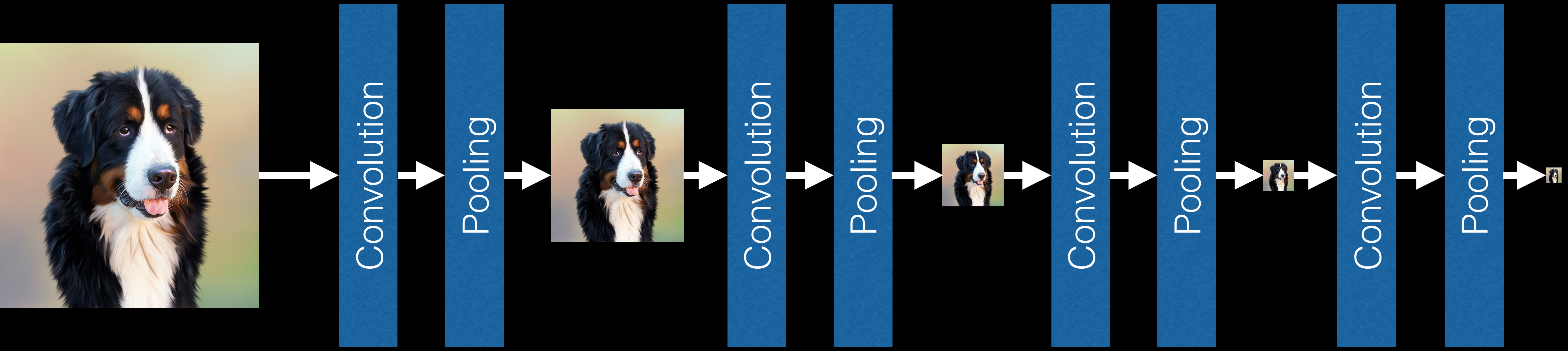
# Layman's understanding

- If we ignore jargons of deep learning,
  - Convolution layer = filtering with zero padding
  - Pooling layer (average) = downsampling

They sound really familiar to graphics researchers!

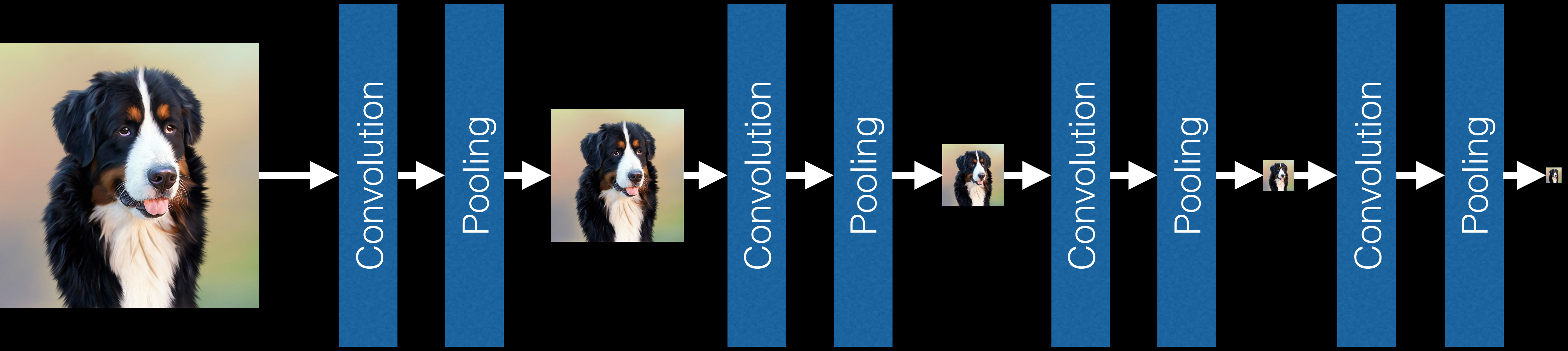

Convolution
Pooling


Convolution
Pooling
Convolution
Pooling



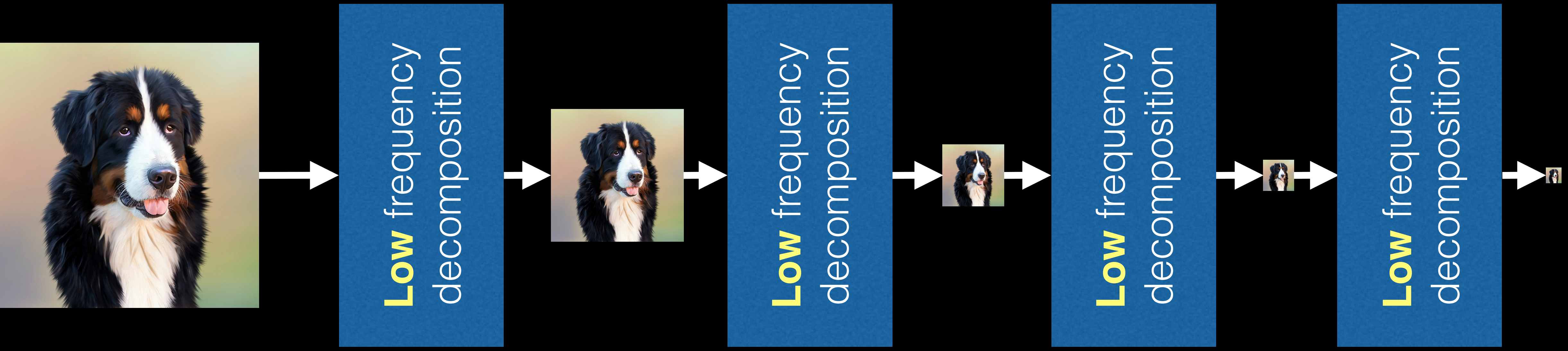
Convolution
Pooling
Convolution
Pooling
Convolution
Pooling
Convolution
Pooling

# Key observation



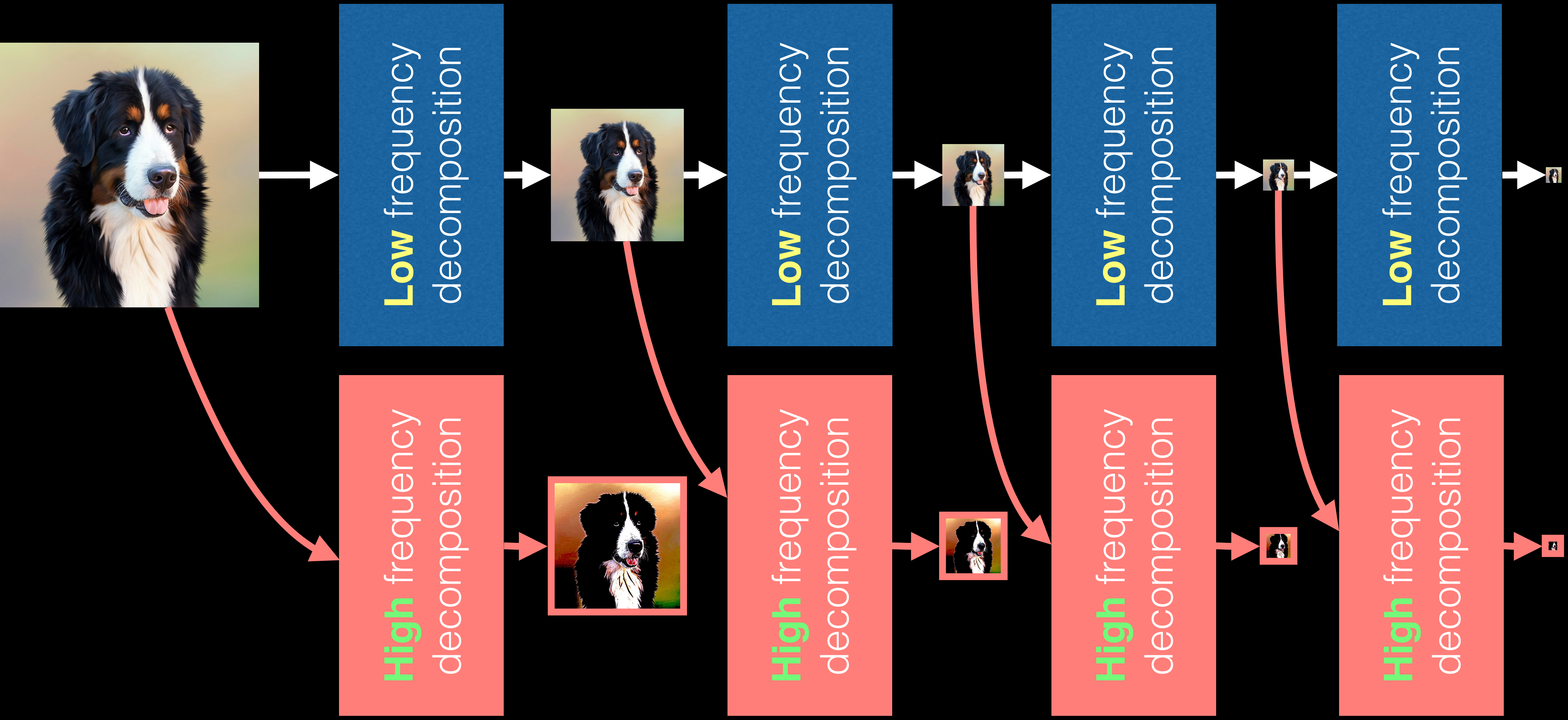
# Key observation

- Convolution followed by pooling is a limited version of multi-resolution analysis via wavelets



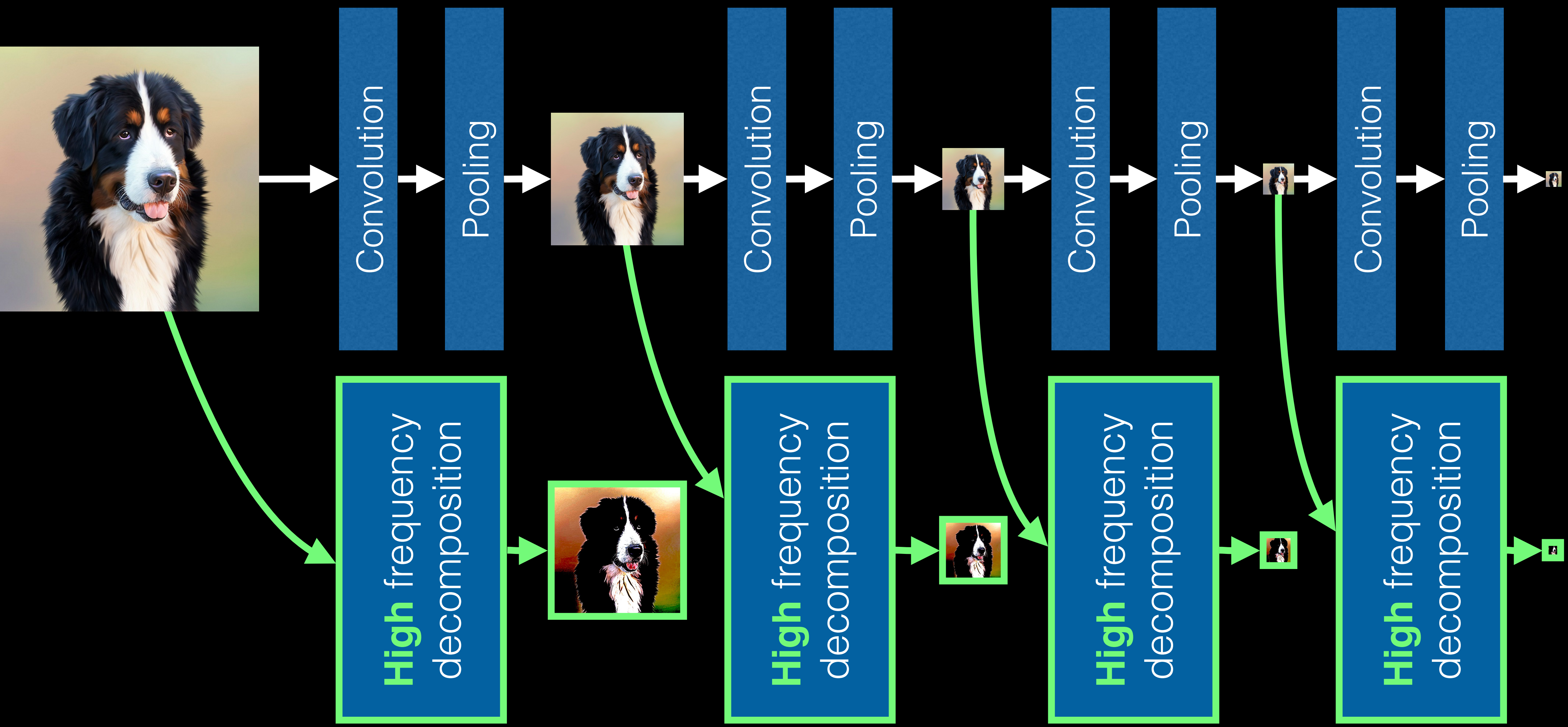
# Key observation

- Convolution followed by pooling is a limited version of multi-resolution analysis via wavelets



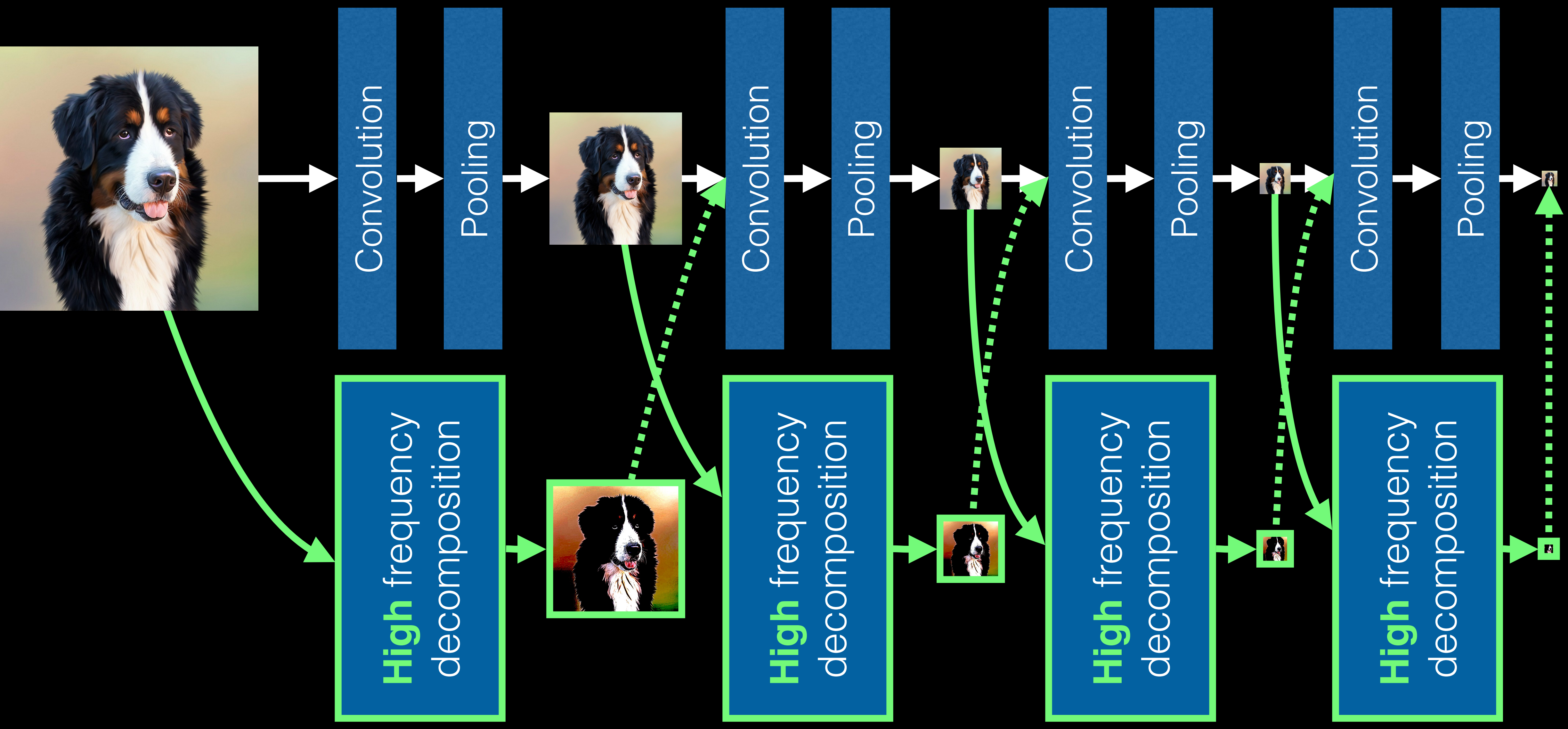
# Wavelet CNNs

- Integrate multi-resolution analysis into CNNs



# Wavelet CNNs

- Integrate multi-resolution analysis into CNNs



# Wavelet CNNs

- Integrate multi-resolution analysis into CNNs
  - Constrain convolution and pooling layers in order to form wavelets
  - Given wavelets, several parameters are fixed (= reduce the number of training parameters)
  - Retain information of the input longer

# Applications

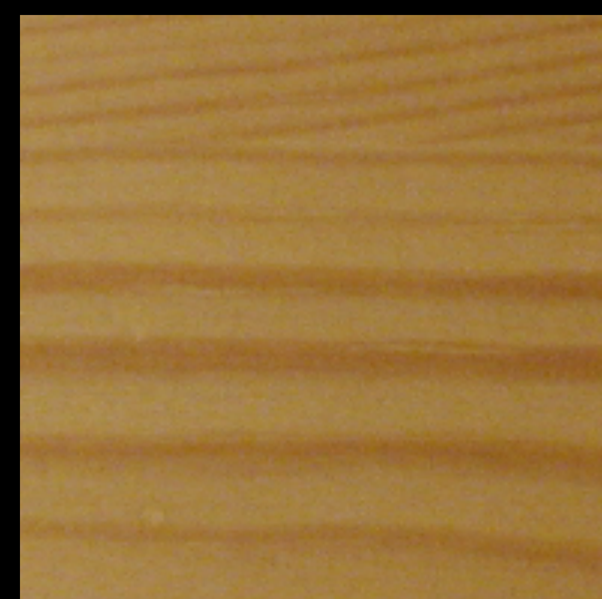
- Wavelet CNN is a general neural network model
- Applied to two difficult tasks even with CNNs
  - Texture classification
  - Image annotation

# Texture classification

- Classify textures into the same materials
- Difficult task even for CNNs due to variation

Aluminium foil

Cork



Wood

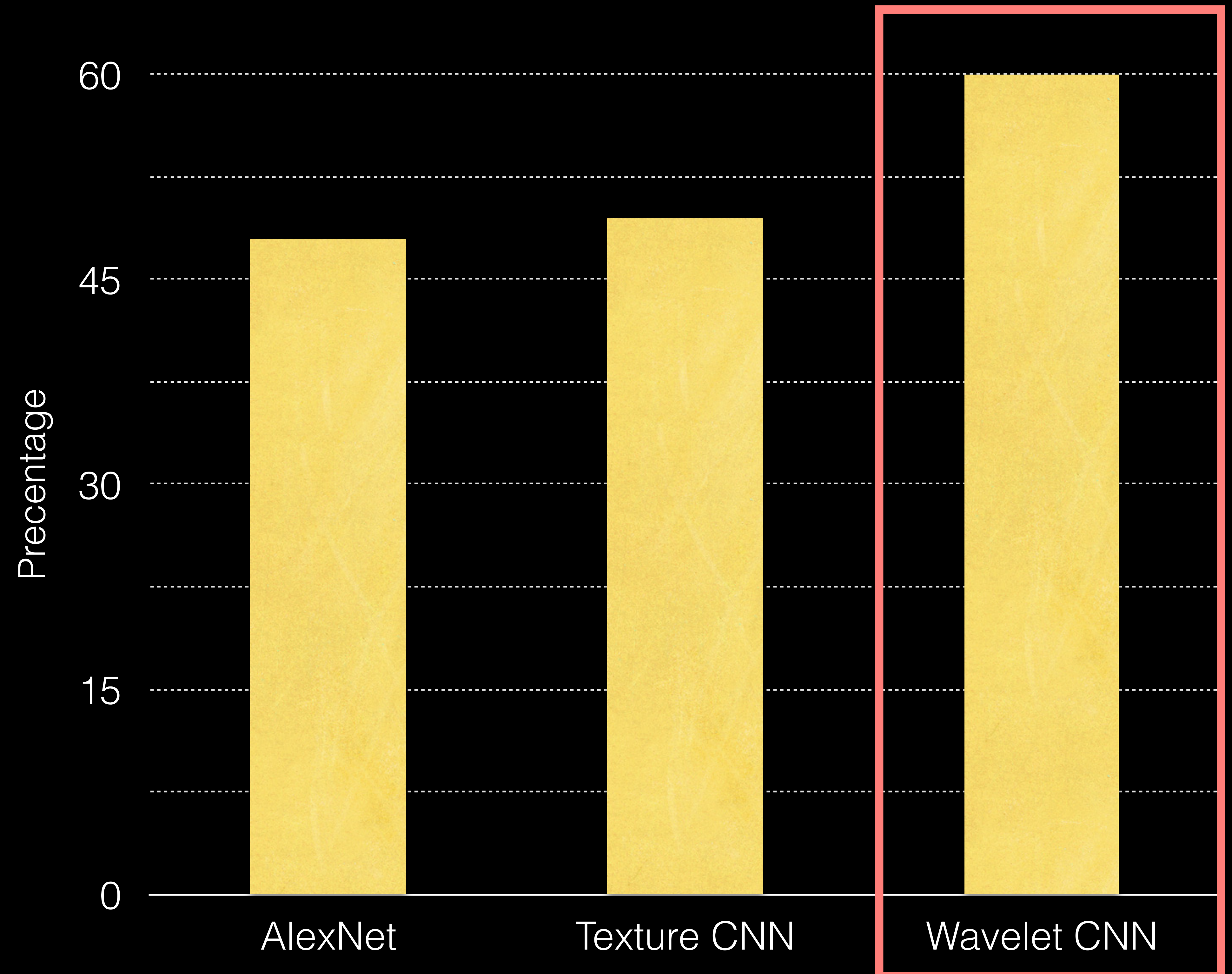
Wool

Images from KTH-TIPS2-b dataset

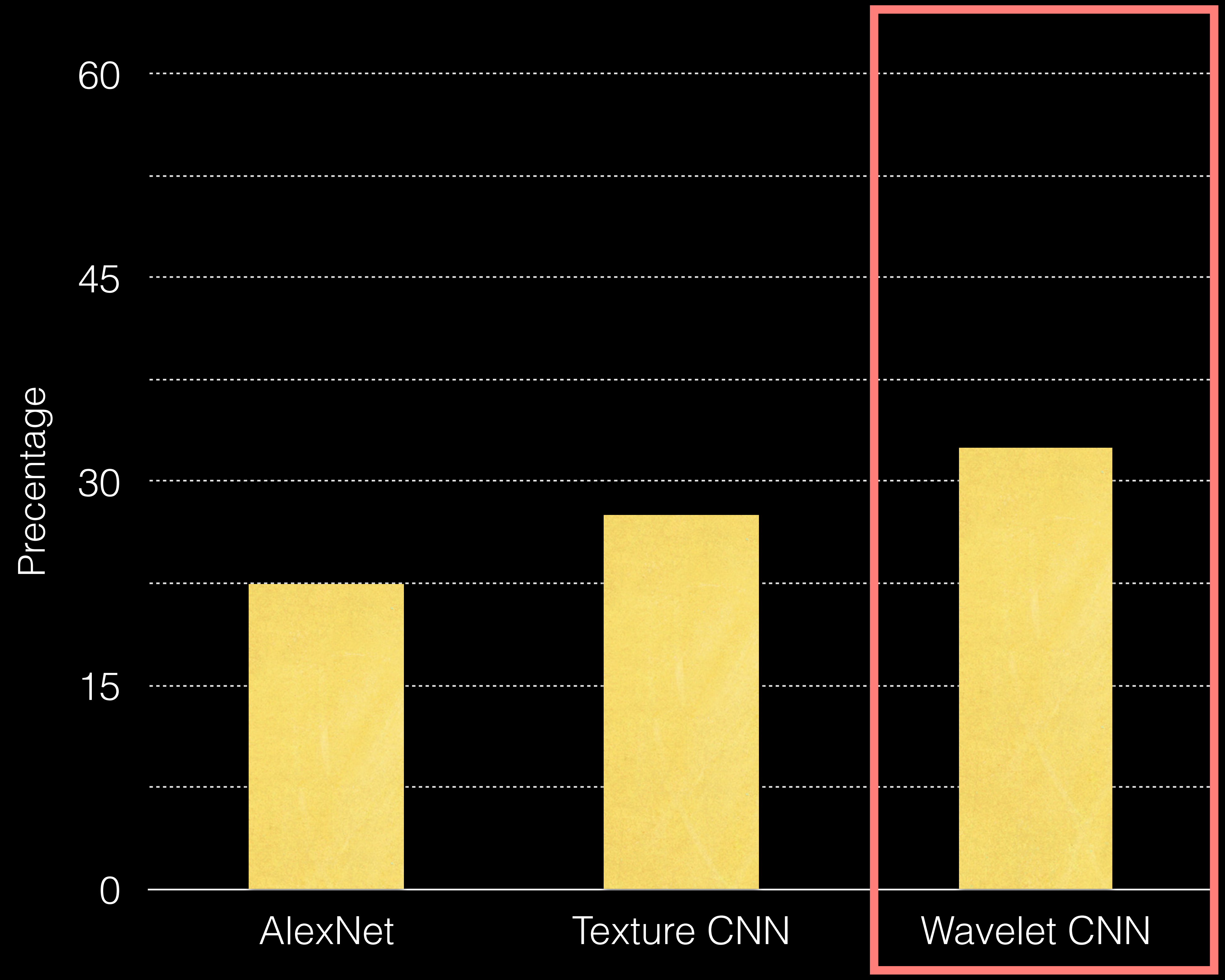
# Experiments

- Two publicly available texture datasets
  - KTH-TIPS2-b: 11 classes of 432 images
  - DTD: 47 classes of 120 images in the wild
- Trained wavelet CNNs and others from scratch

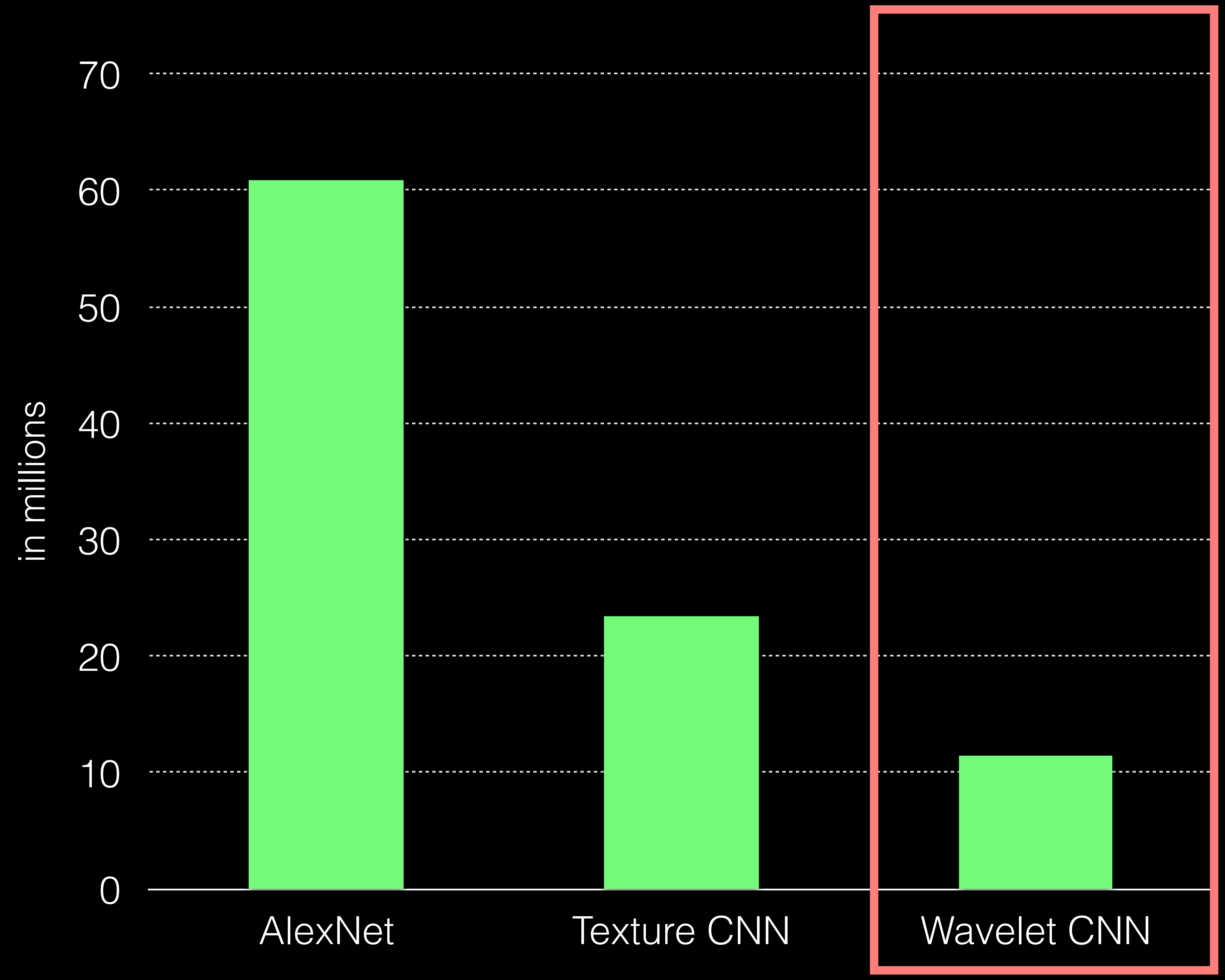
# Accuracy (KTH-TIPS2-b)



# Accuracy (DTD)



# Number of parameters



# Image annotation

- Automatically tag images by words
- Used wavelet CNNs to replace the CNN part

umbrella, cup, dining table, chair, person

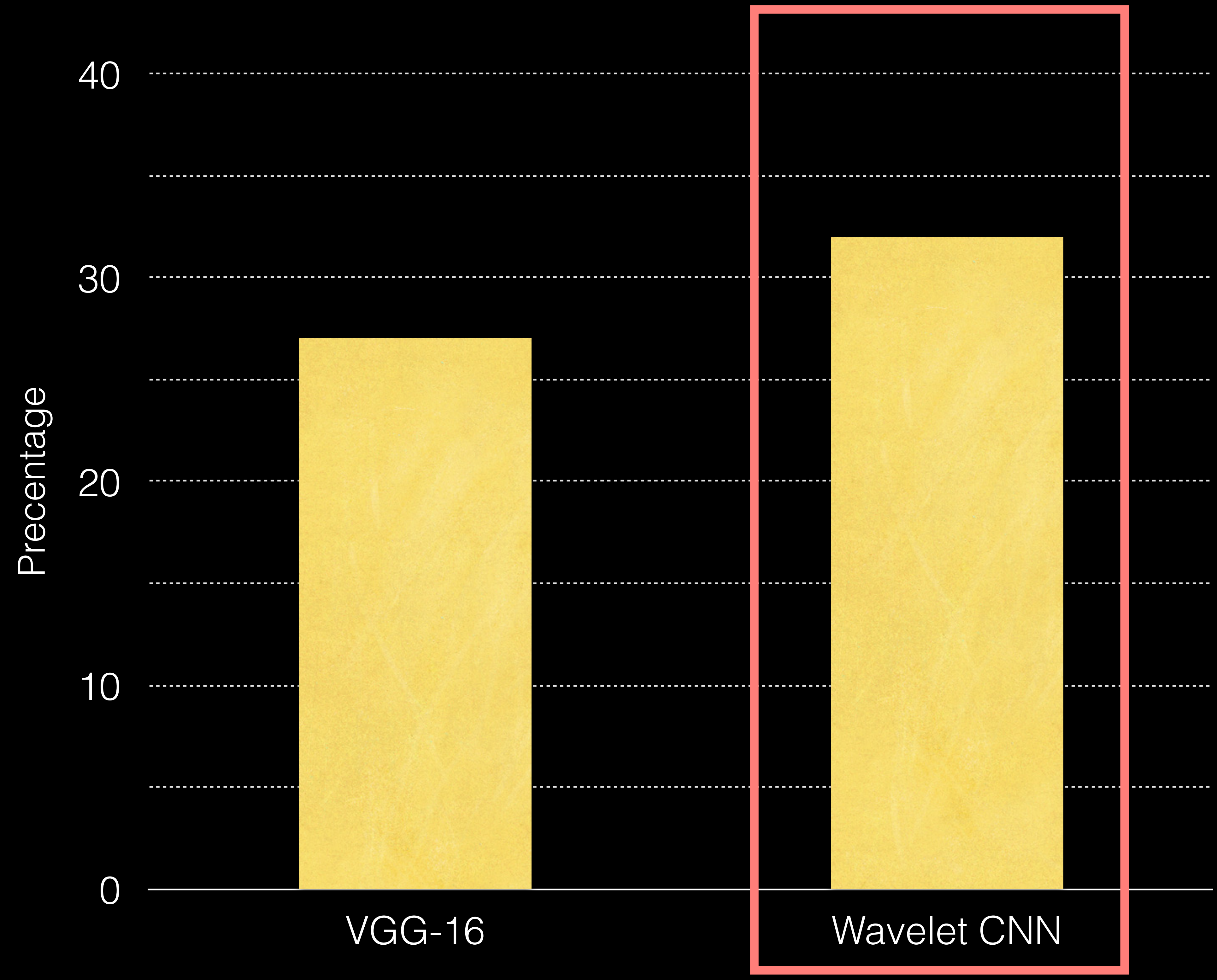
baseball bat, baseball glove, cellphone, person

Feng Liu et al., “Semantic Regularisation for Recurrent Image Annotation”, CVPR, 2017

# Experiments

- Recurrent Image Annotator [Jin et al. 2016]
- Replaced VGG-16 in the model by wavelet CNN
- IAPR-TC12 dataset
  - Vocabulary size: 291
  - Training images: 17665

# Precision



# Number of parameters

# Summary

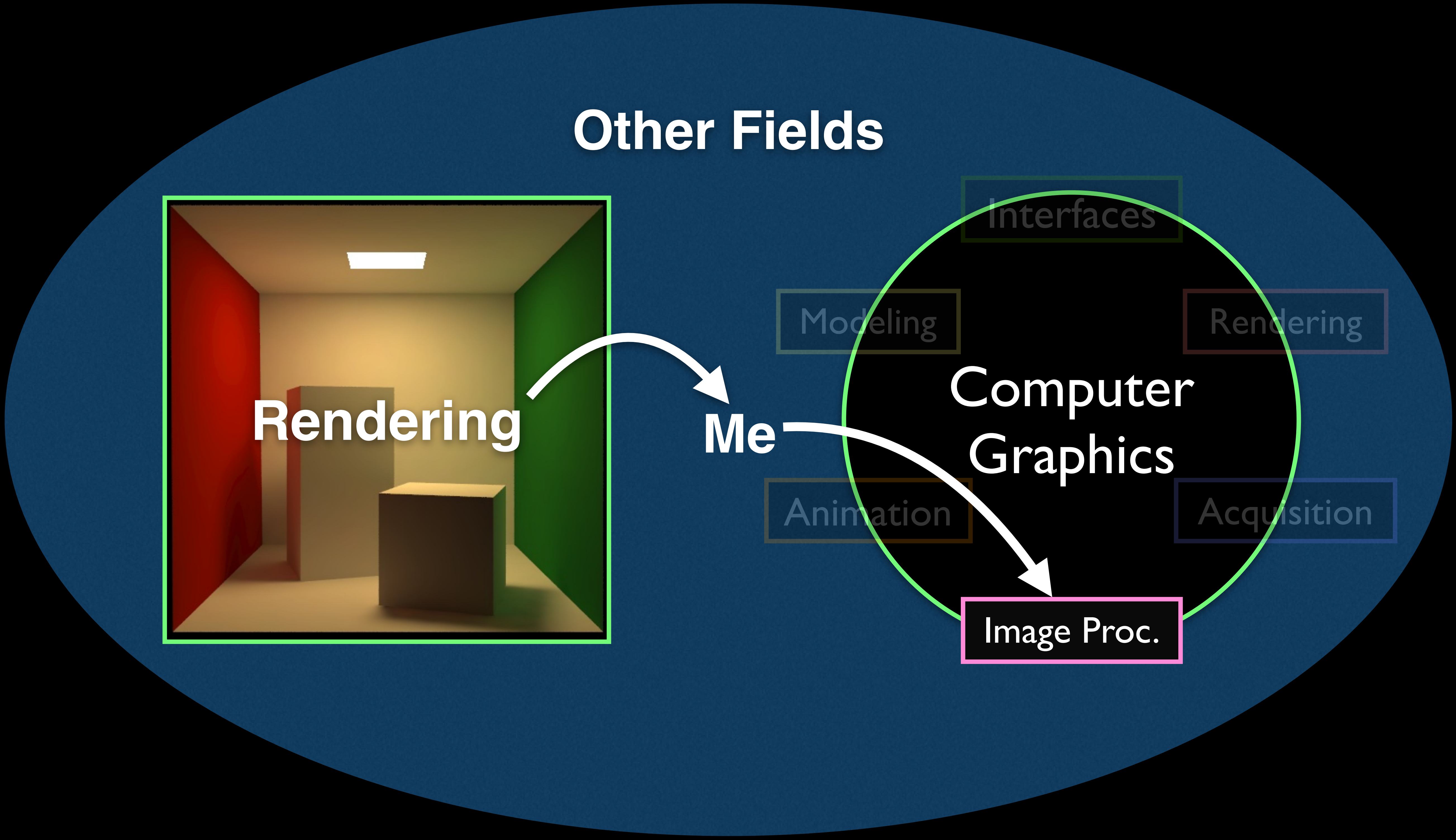
- Layman's view reformulated convolution and pooling in CNNs as wavelet transformation

- Improved results with a smaller number of trainable parameters in two applications

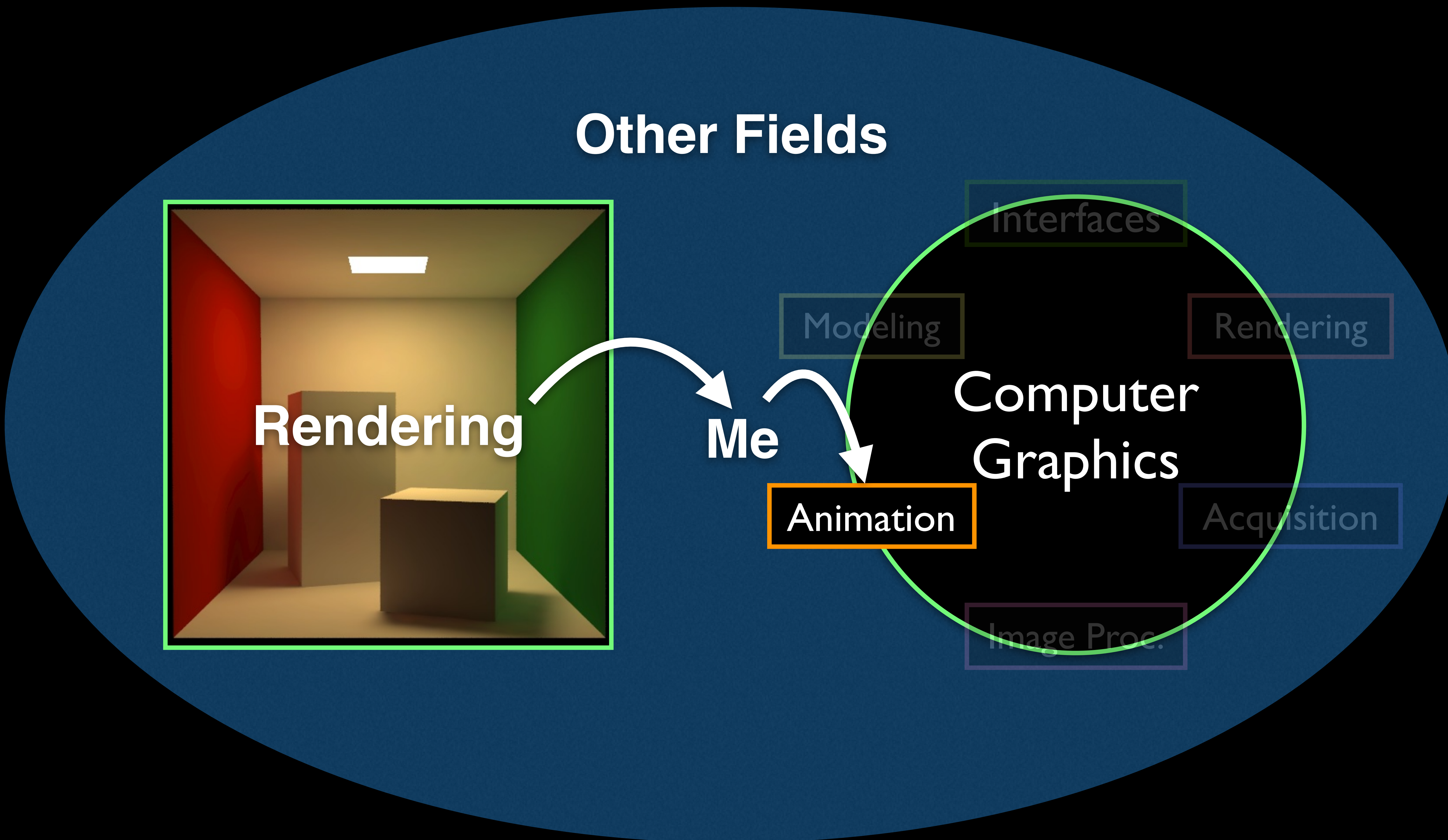
- Applicable to other image processing problems

We are working on making wavelets themselves trainable

# Thinking outside the box



# Thinking outside the box



Animation

# "A Hyperbolic Geometric Flow for Evolving Films and Foams"

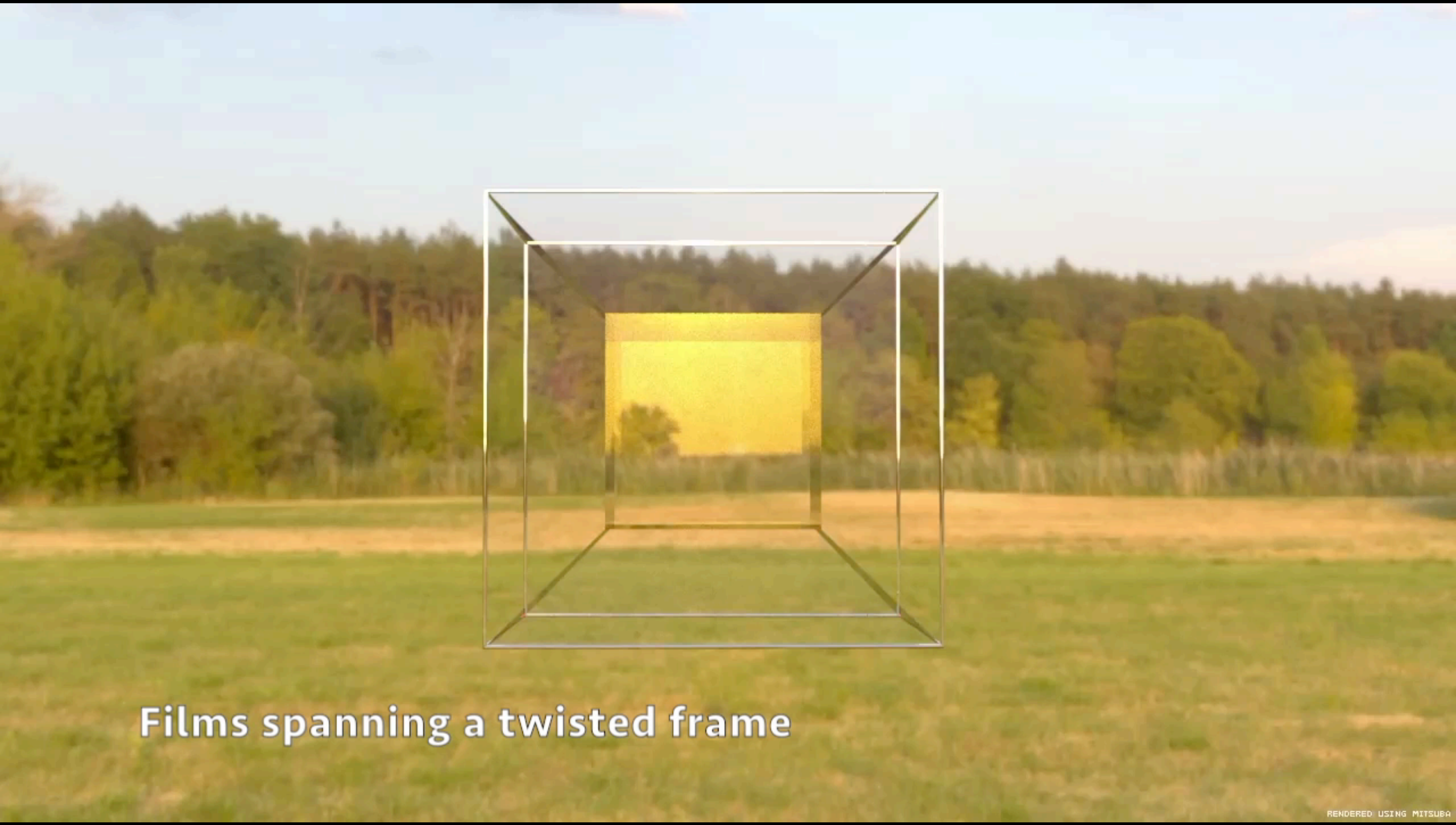
S. Ishida, M. Yamamoto, R. Ando, and T. Hachisuka
ACM Transactions on Graphics (SIGGRAPH Asia 2017), 2017

# Idea

Animation: Physics simulation of soap films

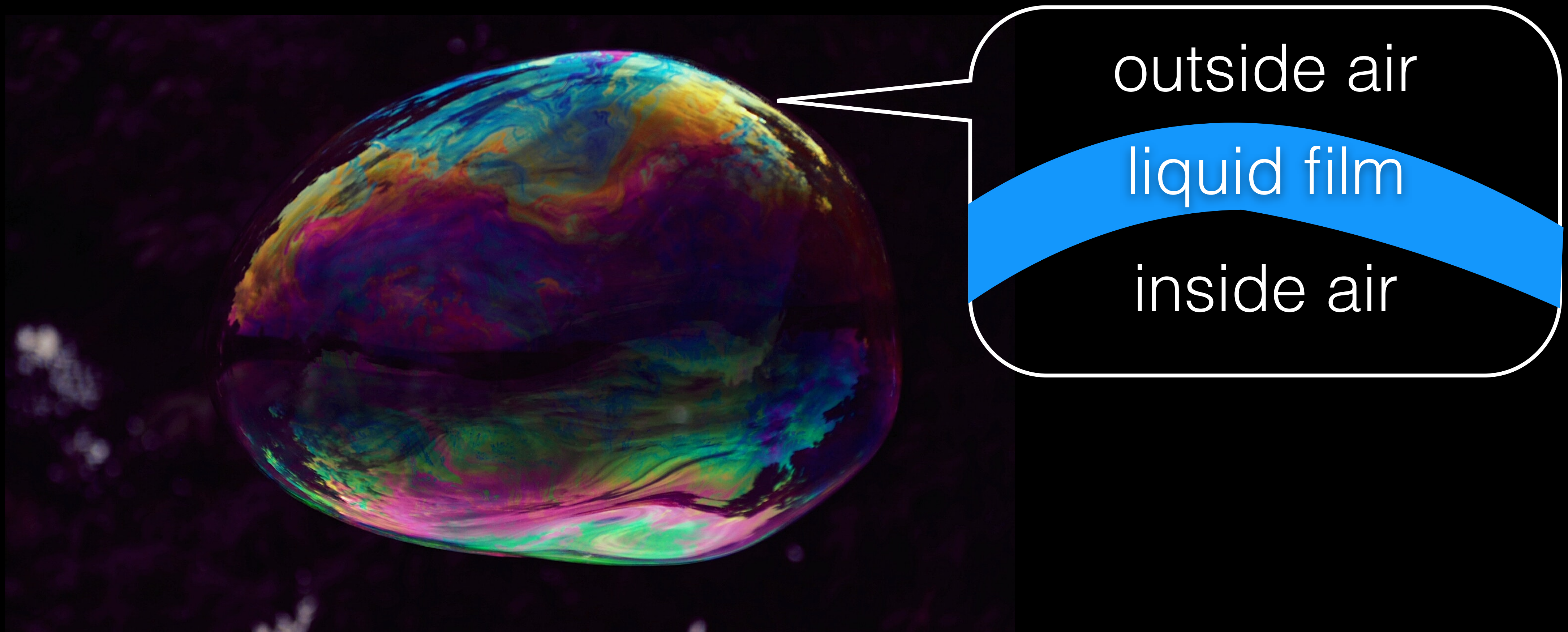
+

Differential geometry: Geometric flow


Films spanning a twisted frame
RENDERED USING MITSUBA

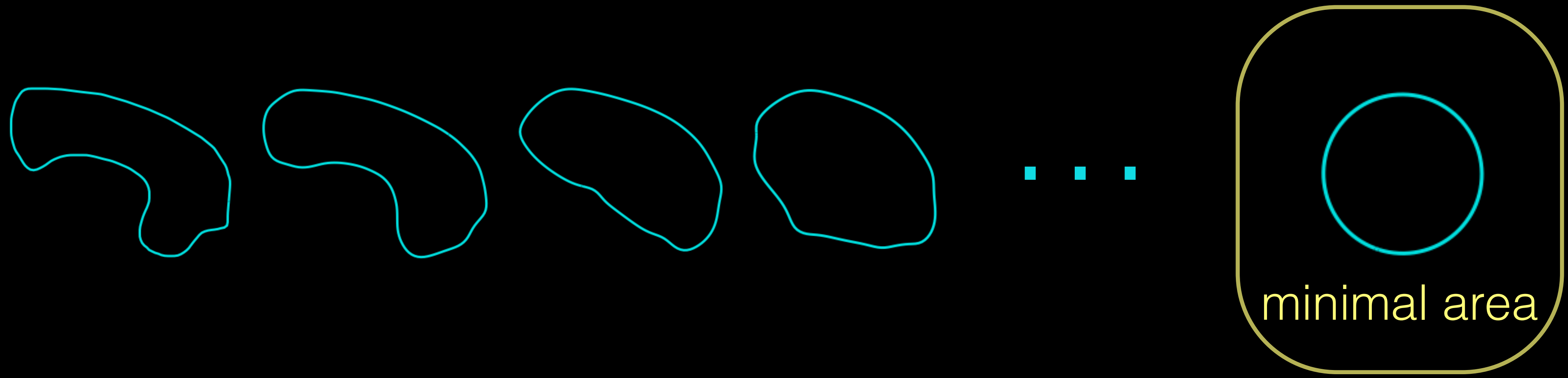
# Liquid films as physics

- Liquid film is extremely thin (~650 nm)
- Coupled dynamics of air and liquid
- Direct simulation via NS equations is difficult



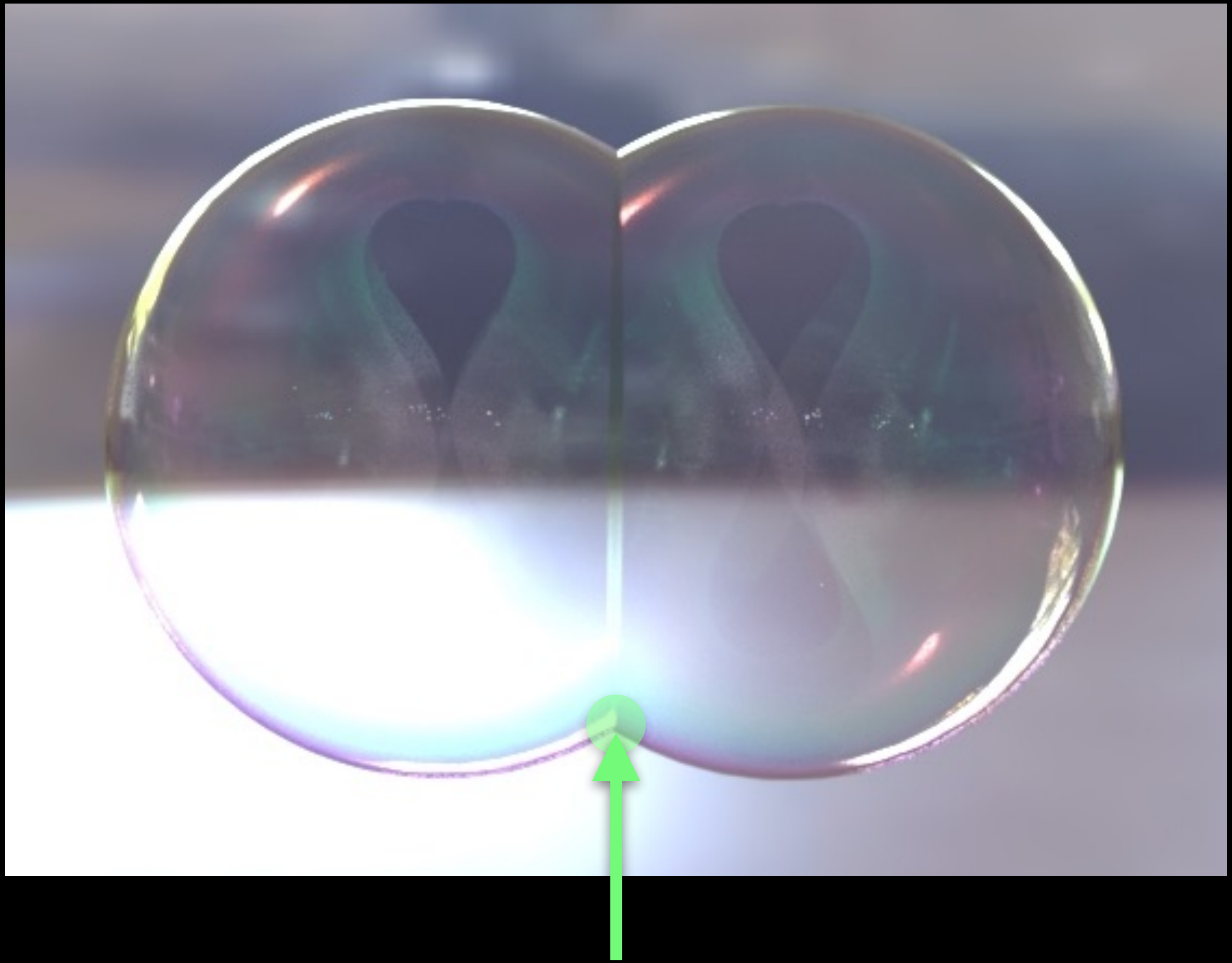
# Liquid films as geometry

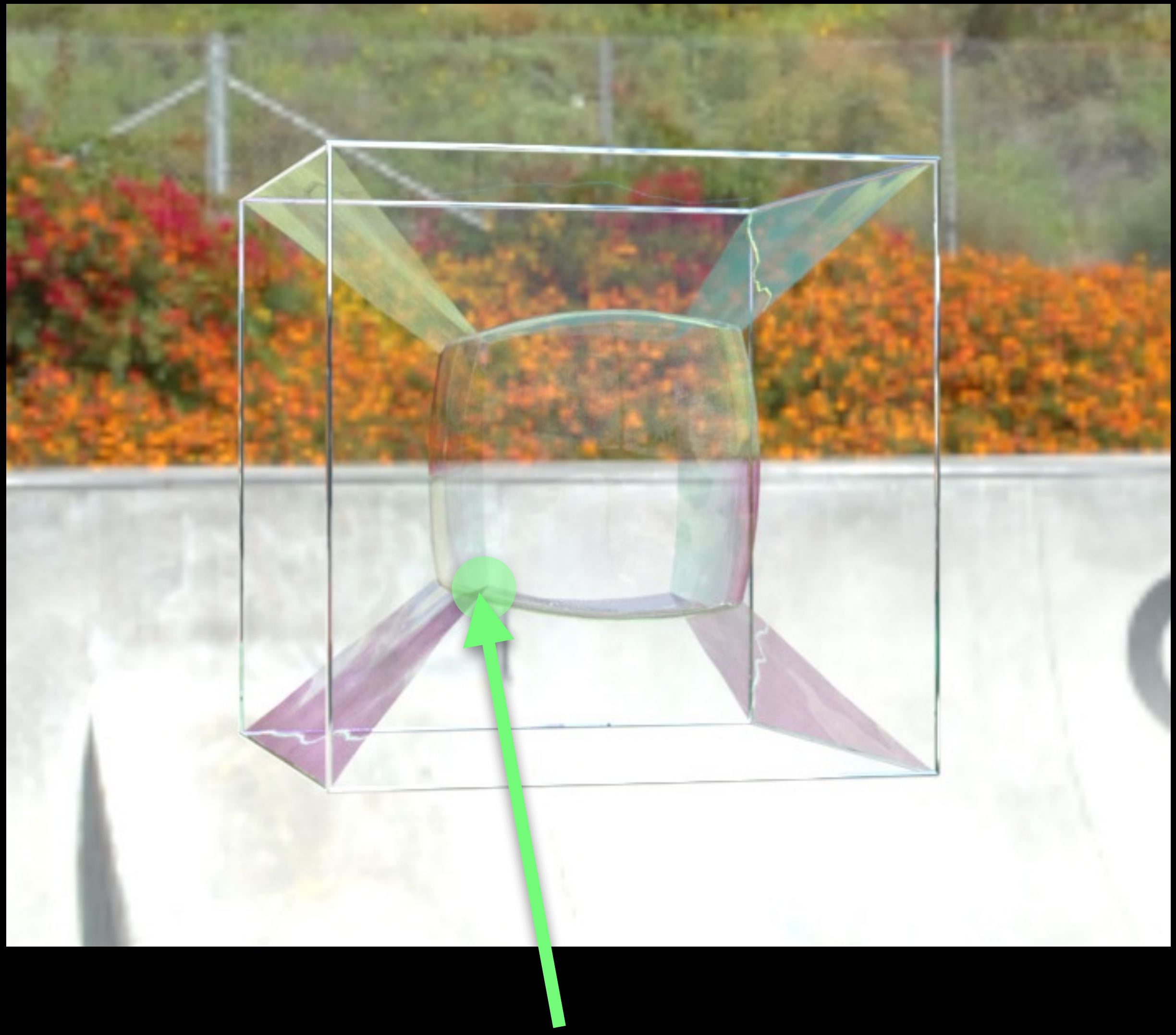
- Subjects of interest in mathematics since 1760
- Steady-state = minimal surface area
- Plateau's law and Plateau's problem



# Plateau's law

- Empirical predictions of steady shapes of films



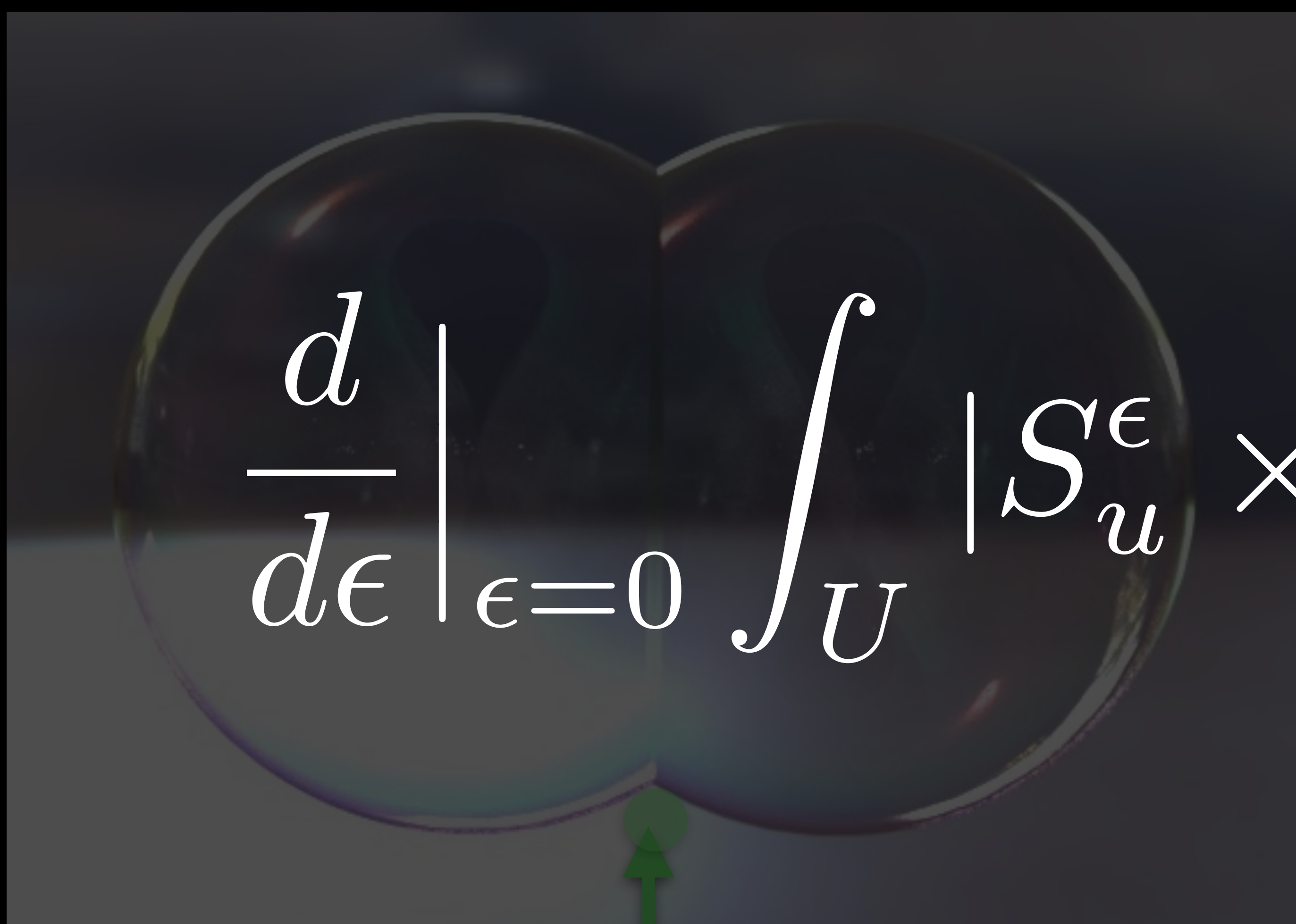


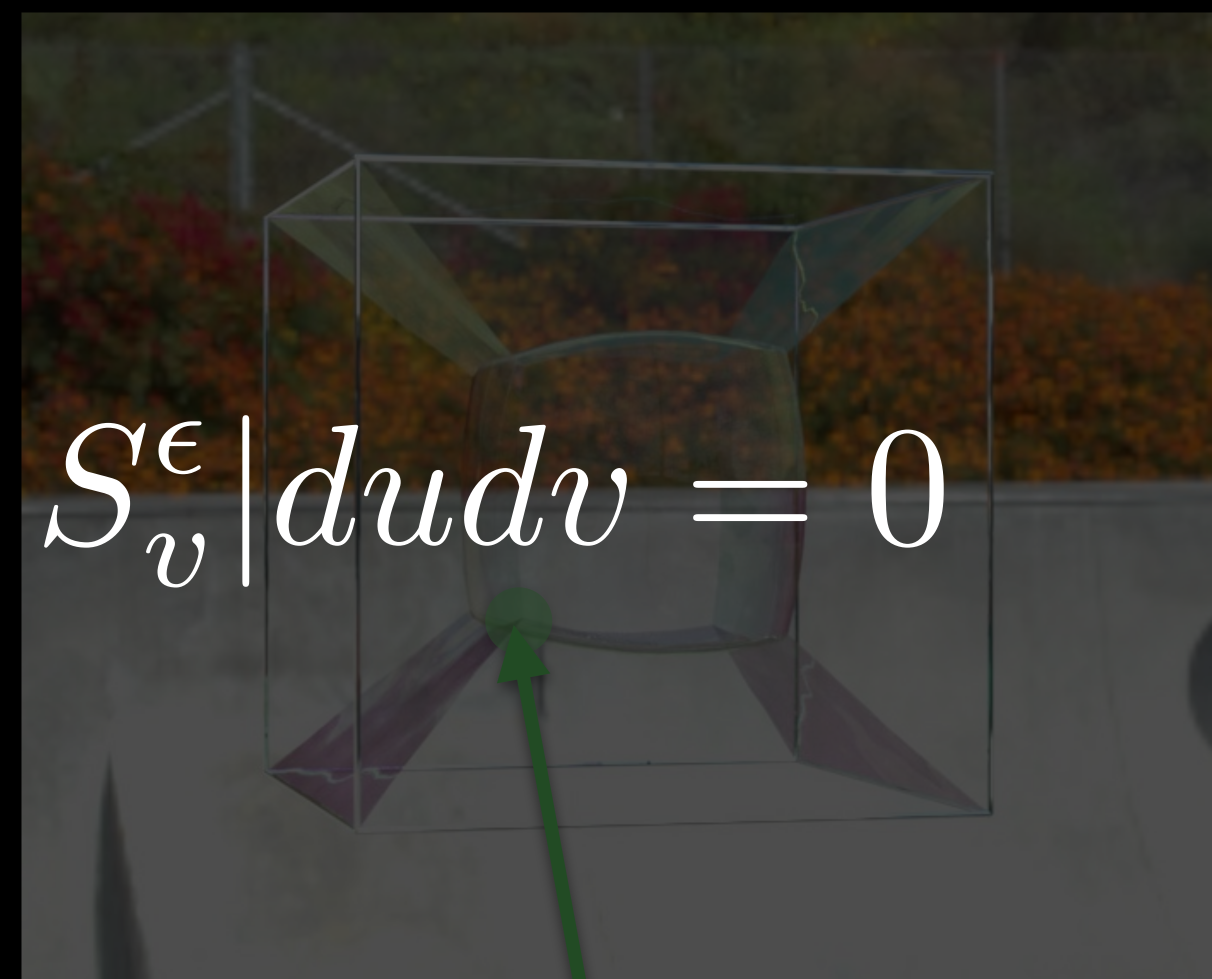
$\arccos(-1/2) = 120^\circ$

$\arccos(-1/3) \approx 109^\circ$

# Plateau’s problem

- Empirical predictions of steady shapes of films



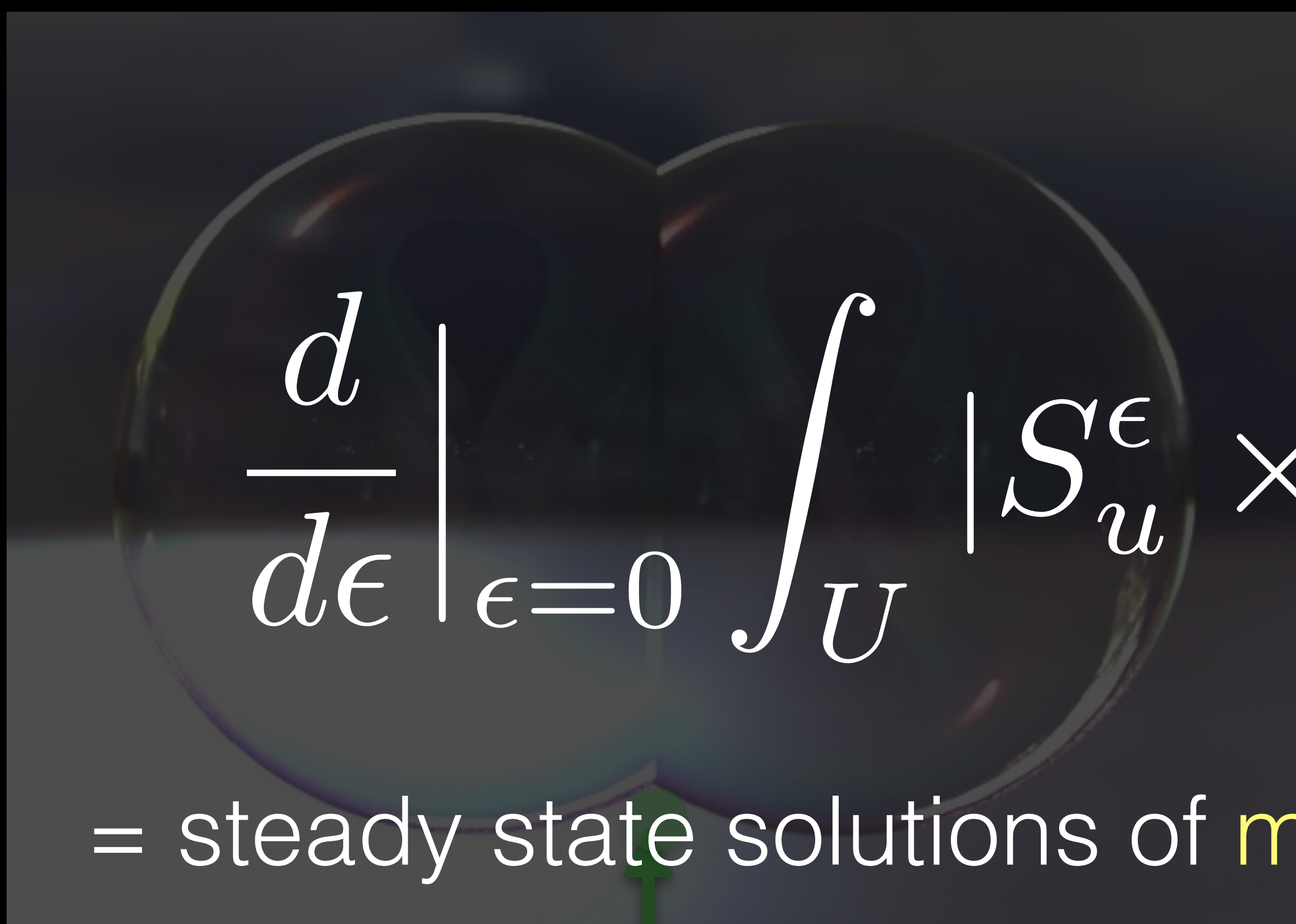


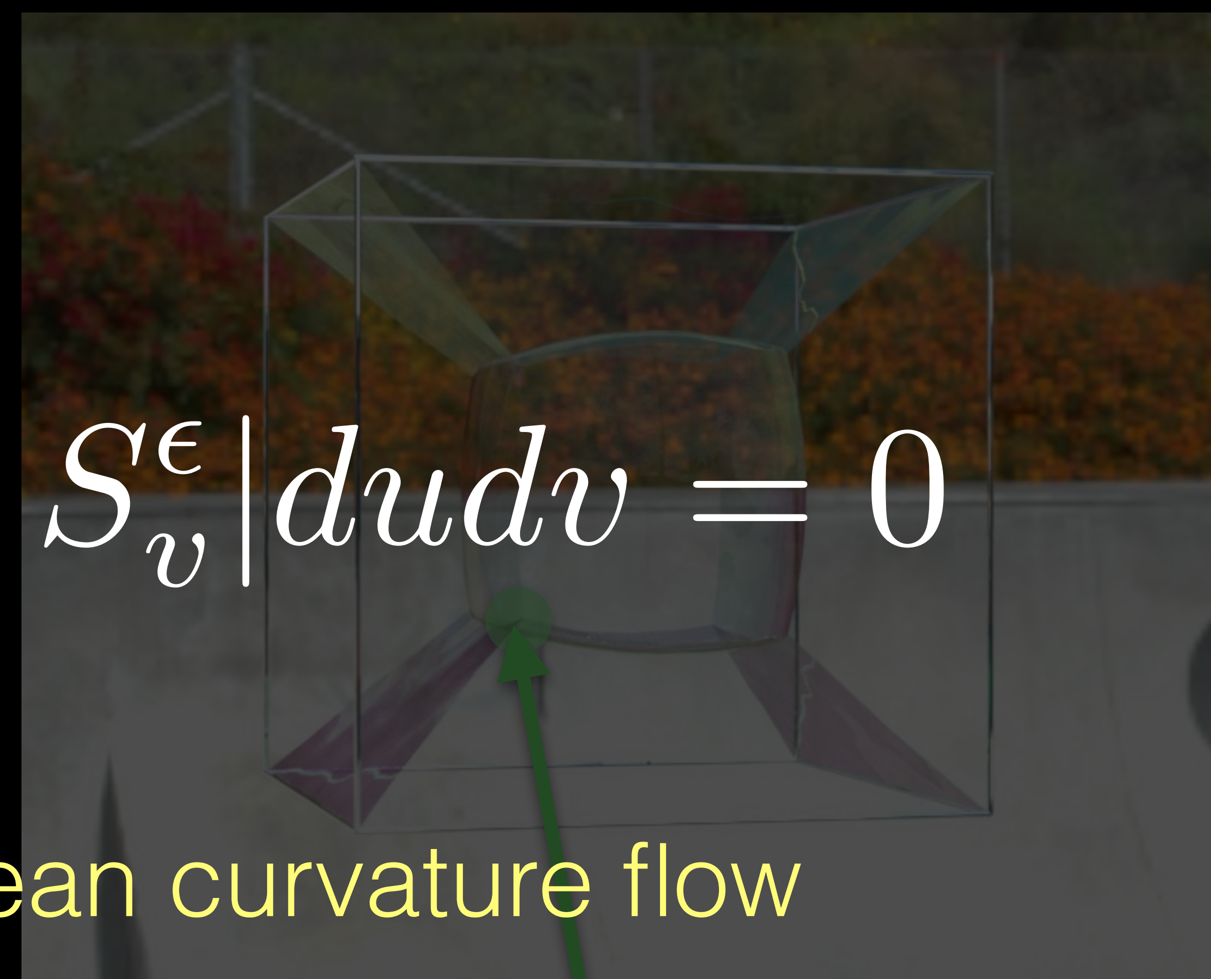
$$\frac{d}{d\epsilon}\Big|_{\epsilon=0} \int_U |S_u^\epsilon \times S_v^\epsilon| du dv = 0$$

$\arccos(-1/2) = 120^\circ$ $\arccos(-1/3) \approx 109^\circ$

# Plateau's problem

- Empirical predictions of steady shapes of films





$$\frac{d}{d\epsilon}\Big|_{\epsilon=0} \int_U |S_u^\epsilon \times S_v^\epsilon| \, du\, dv = 0$$

= steady state solutions of mean curvature flow

$\arccos(-1/2) = 120°$

$\arccos(-1/3) \approx 109°$

# Mean curvature flow (MCF)

- Commonly studied in differential geometry
  - Evolve a surface by its mean curvature
  - Lots of numerical solvers available in graphics

$$\frac{d\boldsymbol{x}}{dt} = -H(\boldsymbol{x}, t)\boldsymbol{n}(\boldsymbol{x}, t)$$

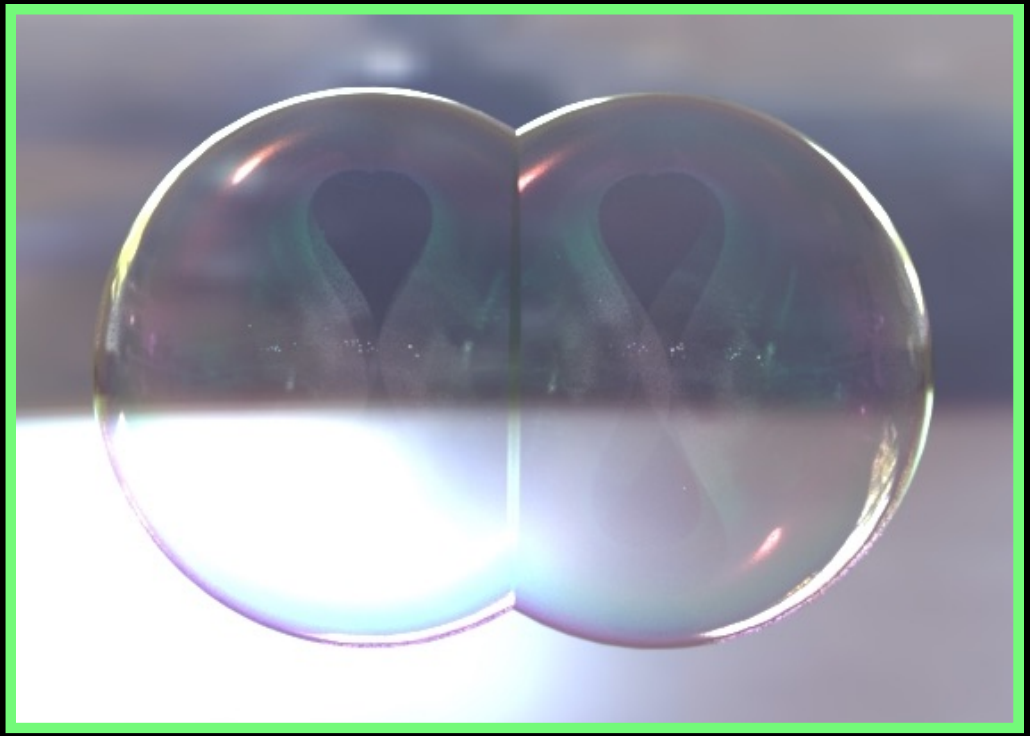
mean curvature     unit normal

Can we just use MCF
to simulate films?

# Geometric flow and film

- Fundamental difference exists
  - Mean curvature flow is a parabolic PDE
  - Film dynamics is a hyperbolic PDE

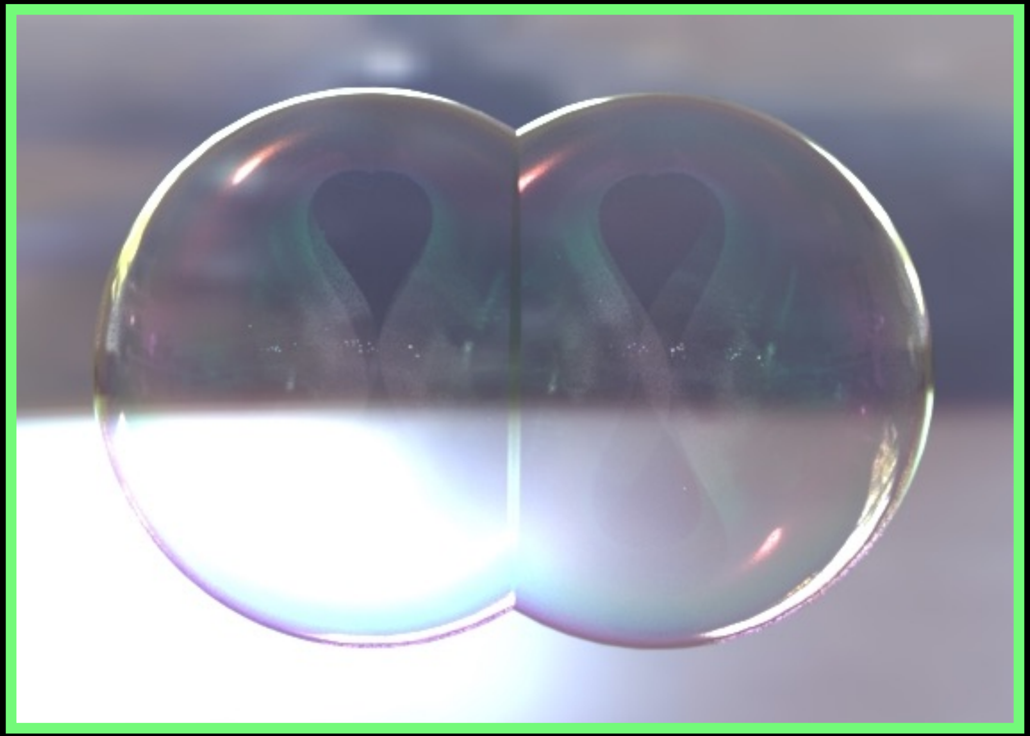
$$\frac{dx}{dt} = -H(x,t)n(x,t) \neq$$



# Hyperbolic MCF and film

- Another geometric flow in differential geometry
  - Hyperbolic MCF is a hyperbolic PDE
  - Film dynamics is a hyperbolic PDE

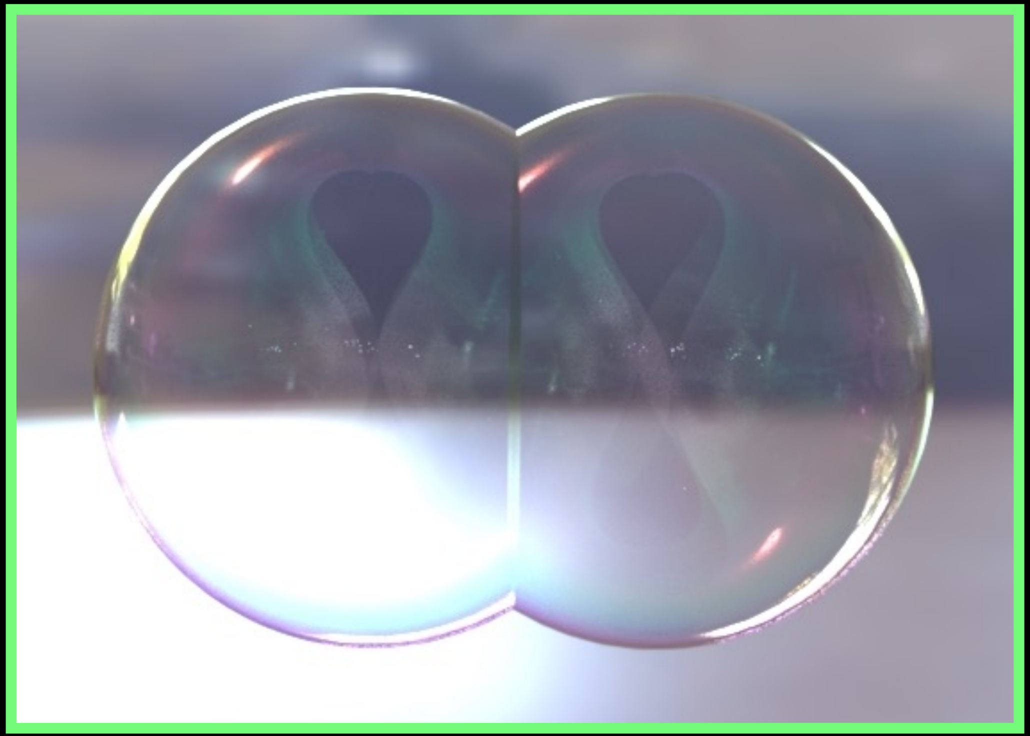
$$\frac{d^2 x}{dt^2} = -H(x,t)n(x,t) \neq$$



# Hyperbolic MCF and film

- Another geometric flow in differential geometry
  - Hyperbolic MCF is a hyperbolic PDE
  - Film dynamics is a hyperbolic PDE

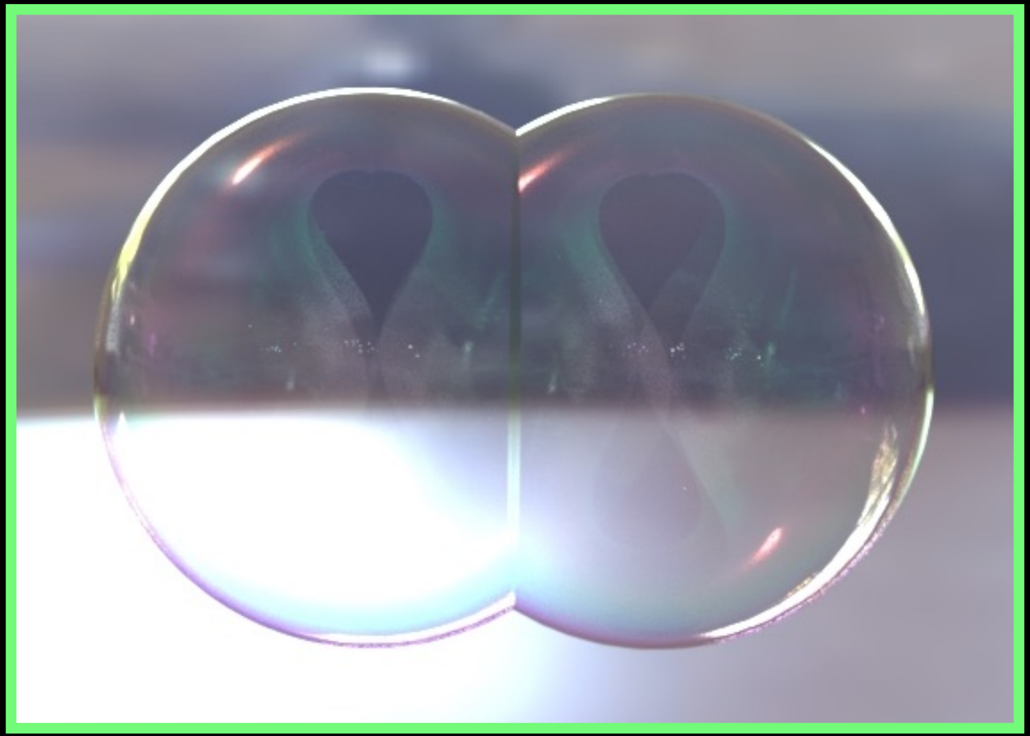
$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) \overset{?}{=}$$



# Hyperbolic MCF and film

- Fundamental difference still exists
  - Hyperbolic MCF is not preserving volume
  - Film dynamics is preserving volume of air

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) \neq$$



# Key observation

**HMCF with vol. preservation = Film dynamics via NS eqn.**

- Steady-state shapes of films can be obtained as solutions of geometric flow
- MCF is a common model, but it’s parabolic
- Hyperbolic MCF doesn’t preserve volume

# HMCF to our model

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t)$$

# HMCF to our model

- No volume preservation

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t)$$

# HMCF to our model

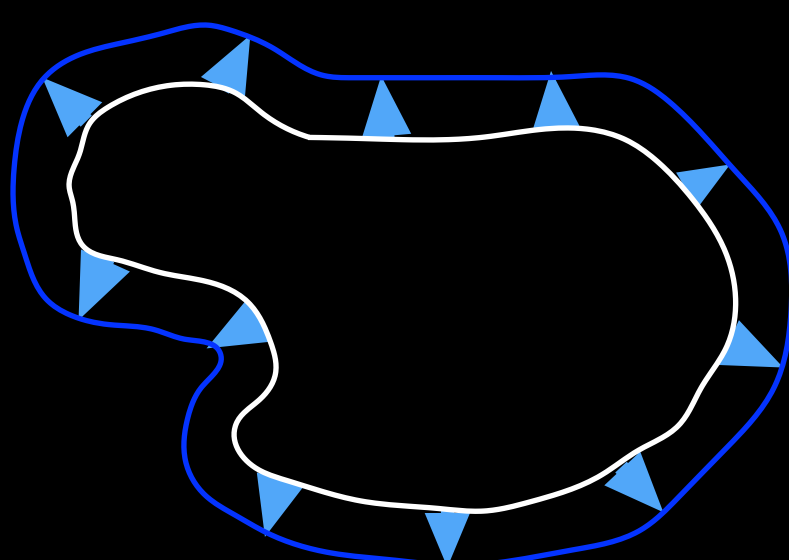
- No volume preservation

Introduce a new pressure term

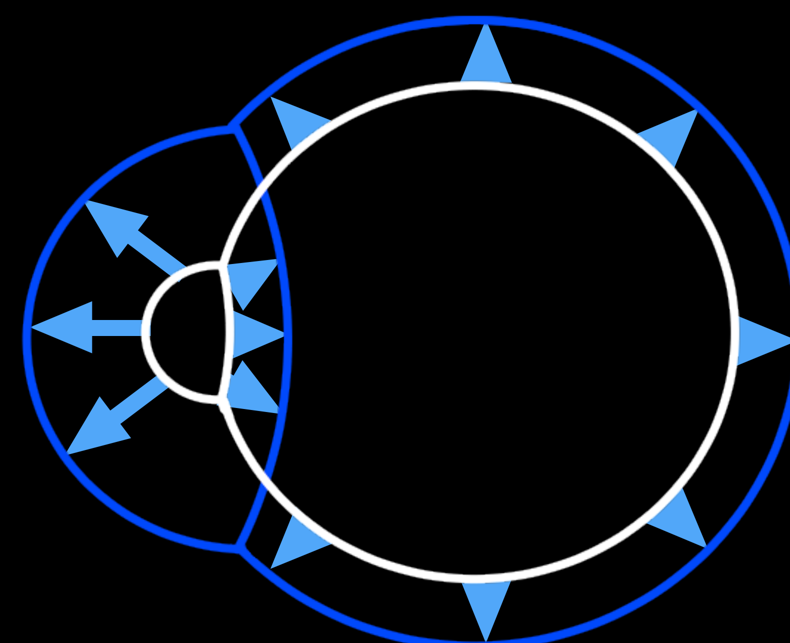
$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) + \Delta p(x,t)n(x,t)$$

# Volume preservation

- Extension of Müller's method for multiple regions
- Pressure in each region is assumed constant



Müller [2009]



Ours

# HMCF to our model

- No volume preservation

Introduce a new pressure term

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) \boxed{+\Delta p(x,t)n(x,t)}$$

# HMCF to our model

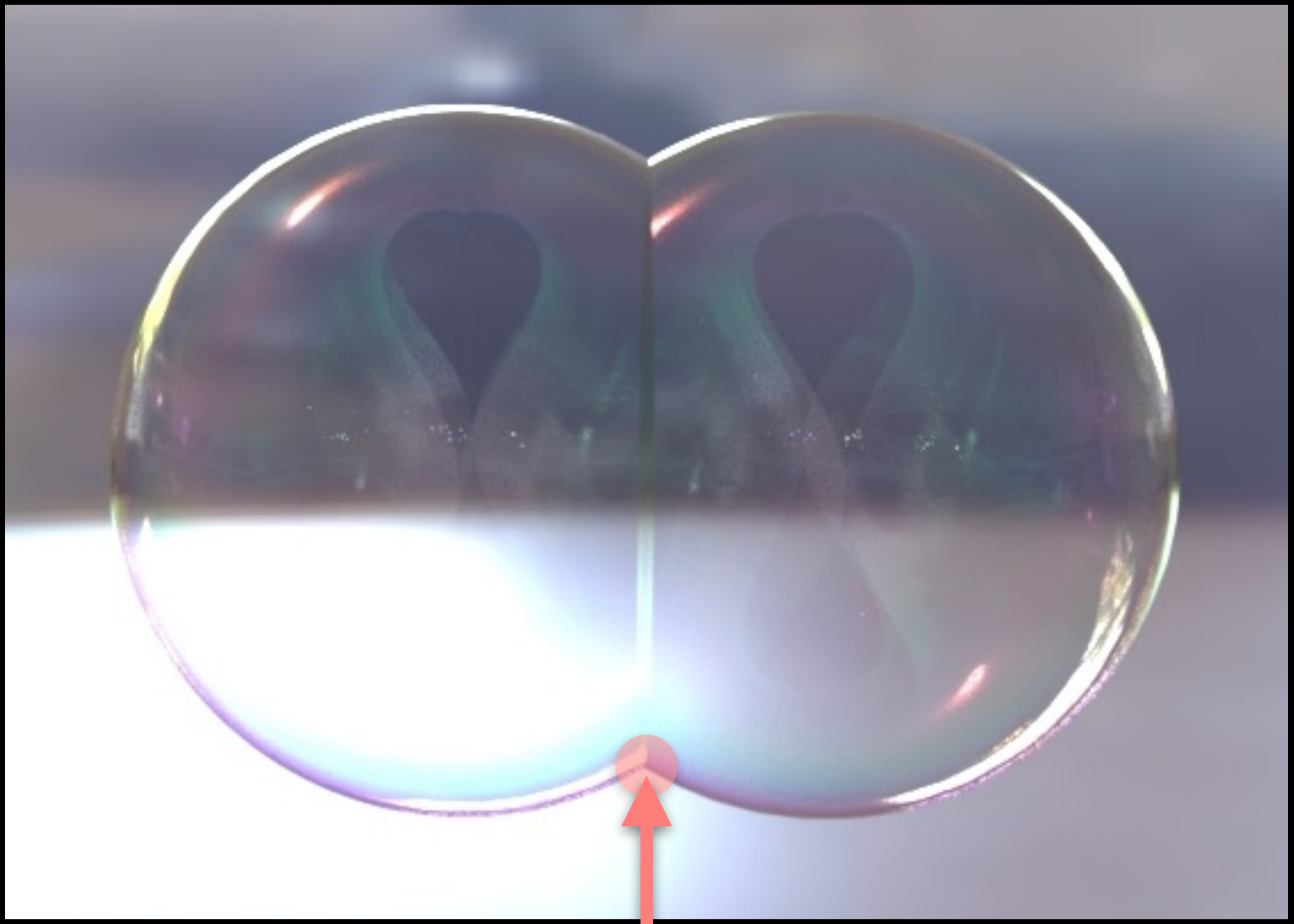
- No volume preservation

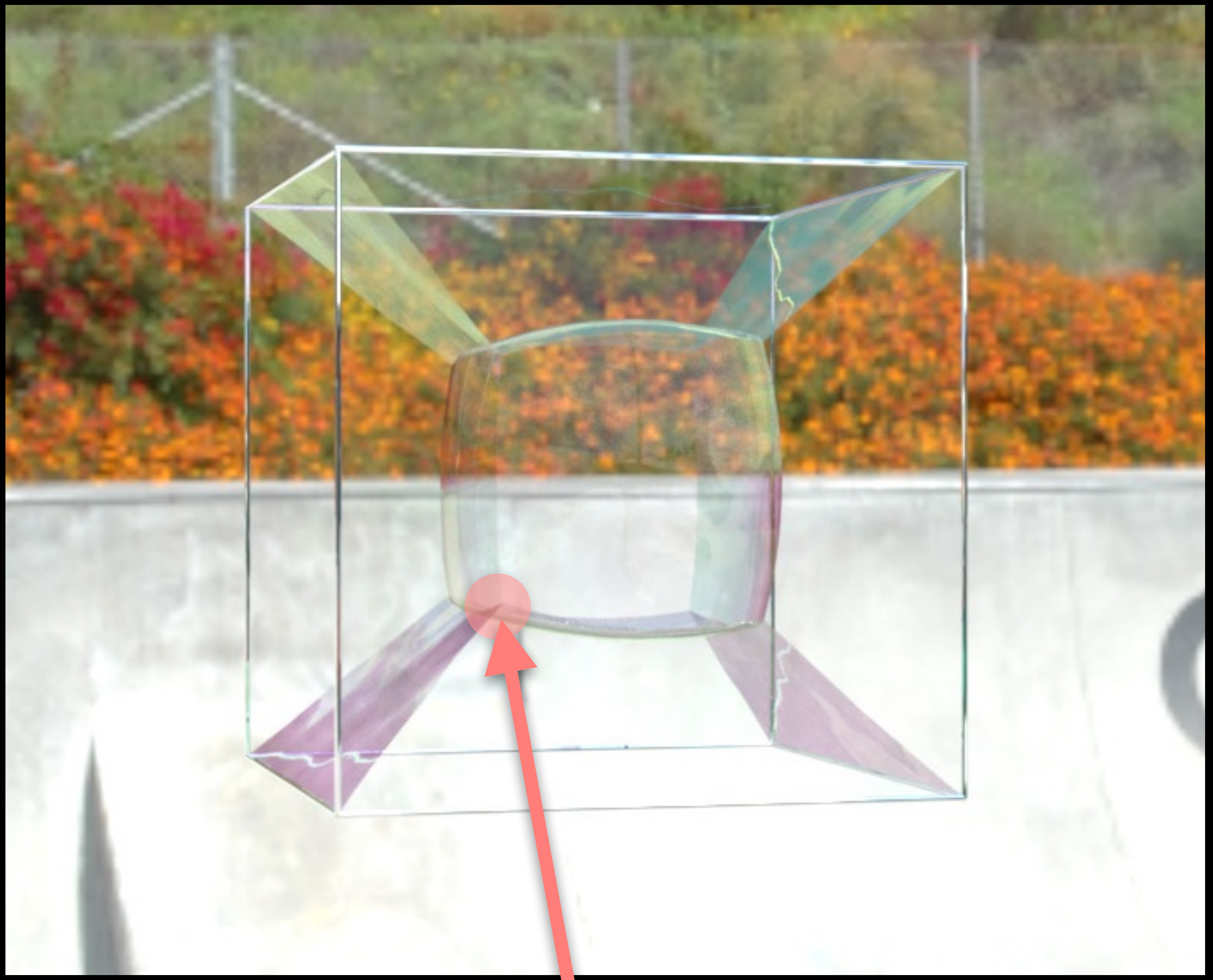
Introduce a new pressure term

- Mean curvature can be undefined

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) \boxed{+\Delta p(x,t)n(x,t)}$$

# Undefined mean curvature





**Undefined exactly on those important points!**

# HMCF to our model

- No volume preservation

Introduce a new pressure term

- Mean curvature can be undefined

$$\frac{d^2x}{dt^2} = -H(x,t)n(x,t) + \Delta p(x,t)n(x,t)$$

# HMCF to our model

- No volume preservation

Introduce a new pressure term

- Mean curvature can be undefined

Replace it by variational derivative

$$\frac{d^2x}{dt^2} = -\frac{\partial A}{\partial x} + \Delta p(x,t) n(x,t)$$

# Variational derivative

- Properties of $\frac{\partial A}{\partial x}$ and $Hn$ match well
  - Direction - maximizes the local area
  - Magnitude - difference from the maximum
- Indeed, $\frac{\partial A}{\partial x} = Hn$ when mean curvature is defined

(proof is in our paper)

# Our model

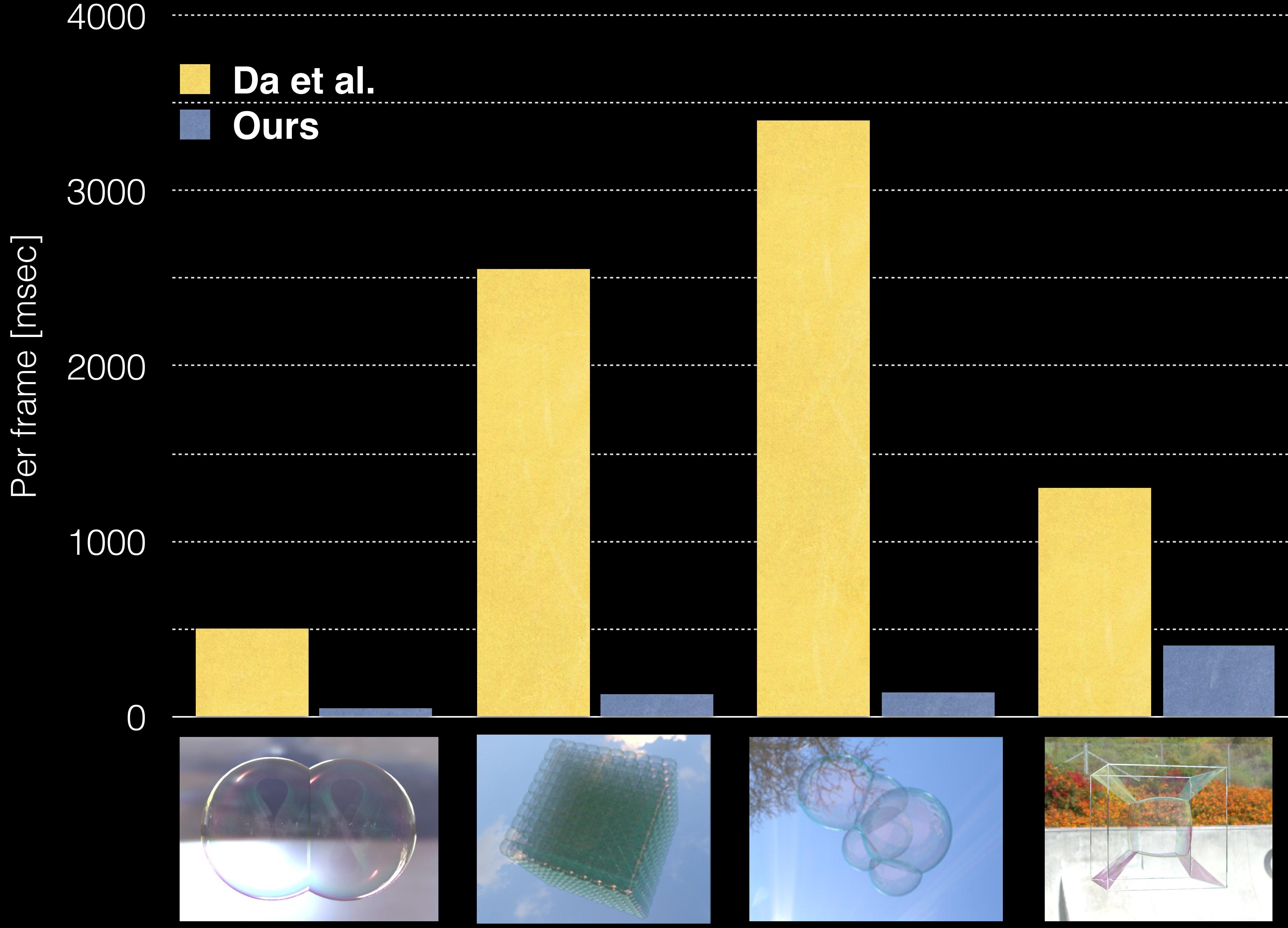
- Preserve volume of trapped air
- Works fine without mean curvature
- NS equations with assumptions become our model

$$\frac{d^2x}{dt^2} = -\beta\frac{\partial A}{\partial x} + \Delta p(x,t)n(x,t)$$

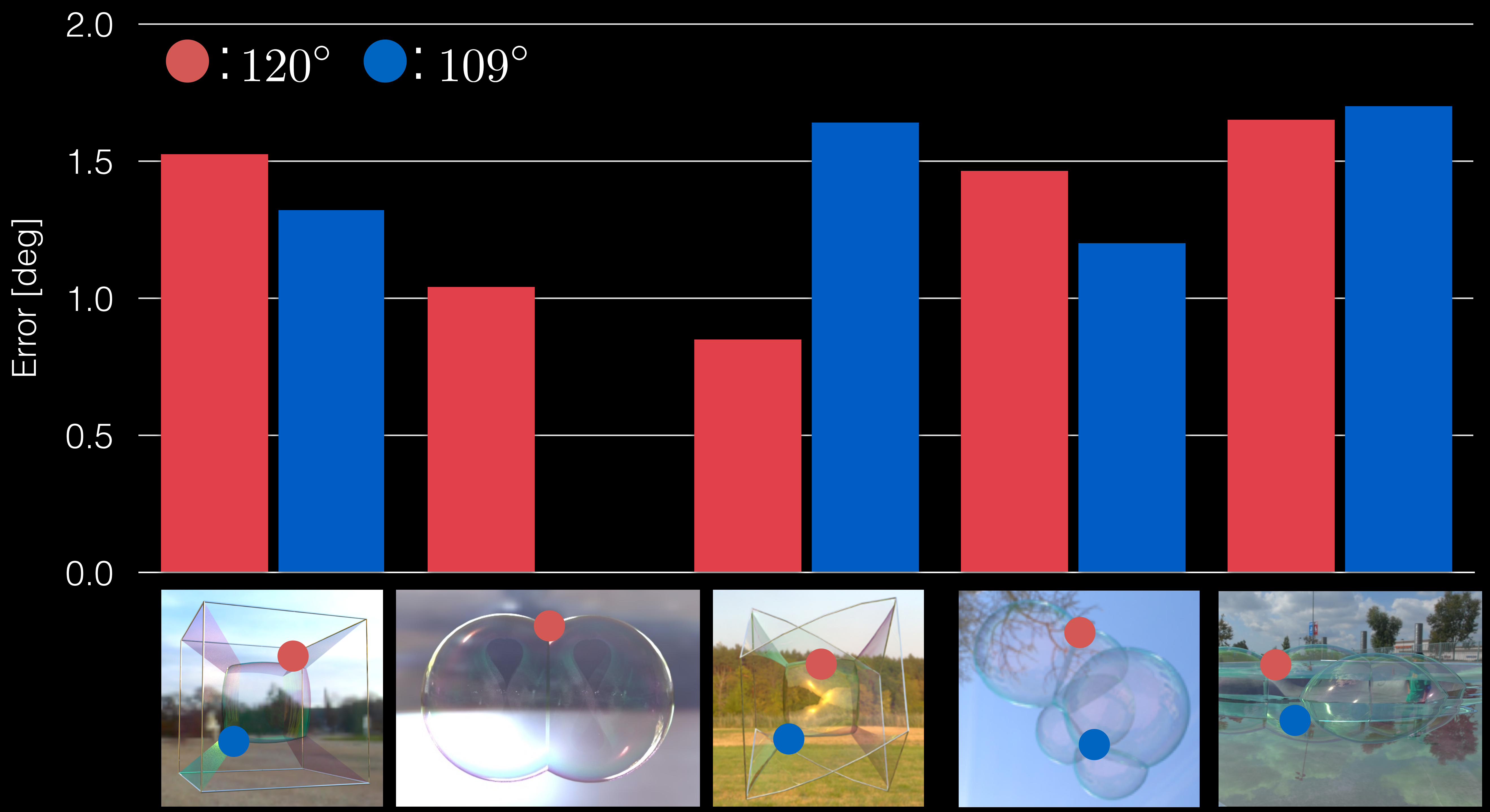
surface tension constant

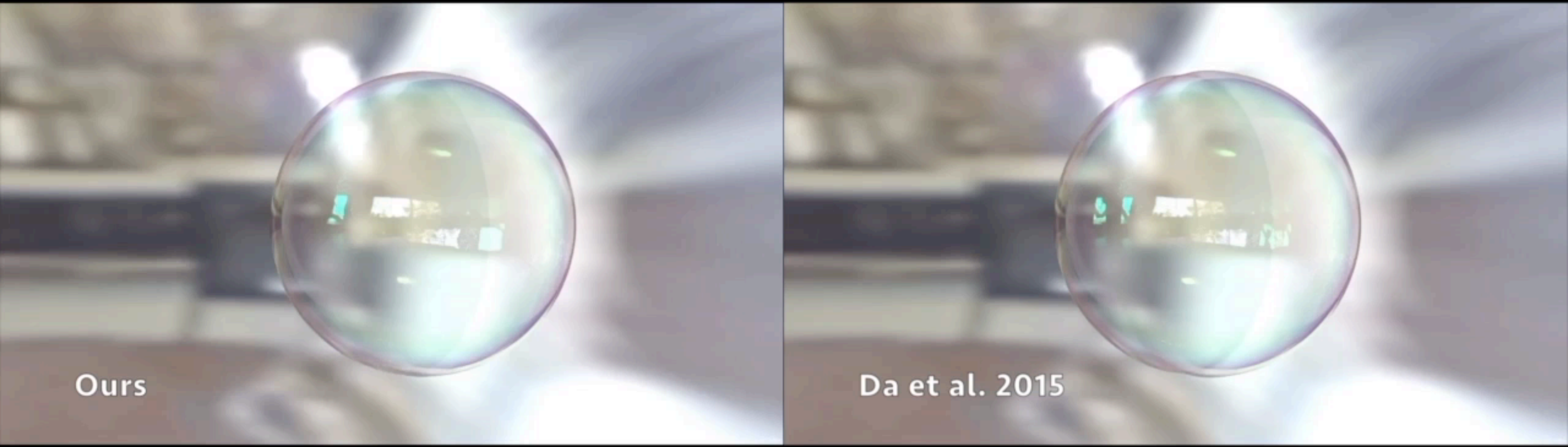
# Results

# Computation cost



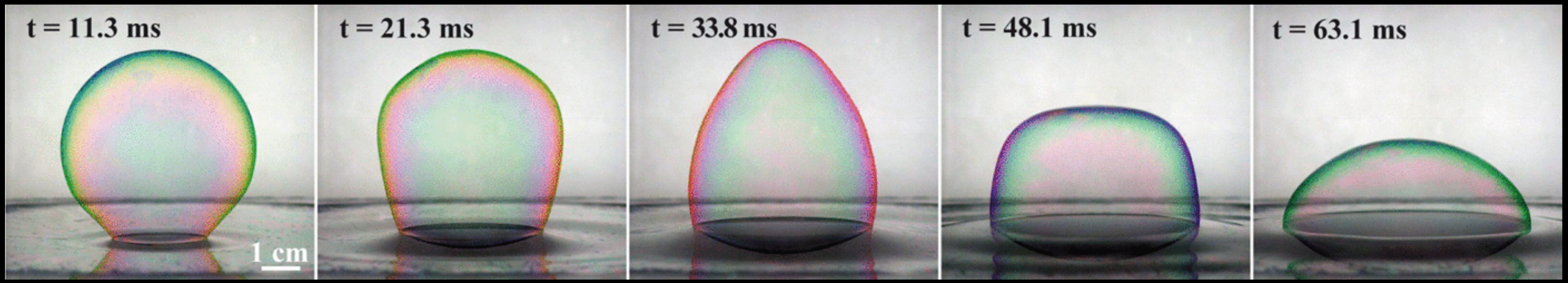
# Accuracy on Plateau's law



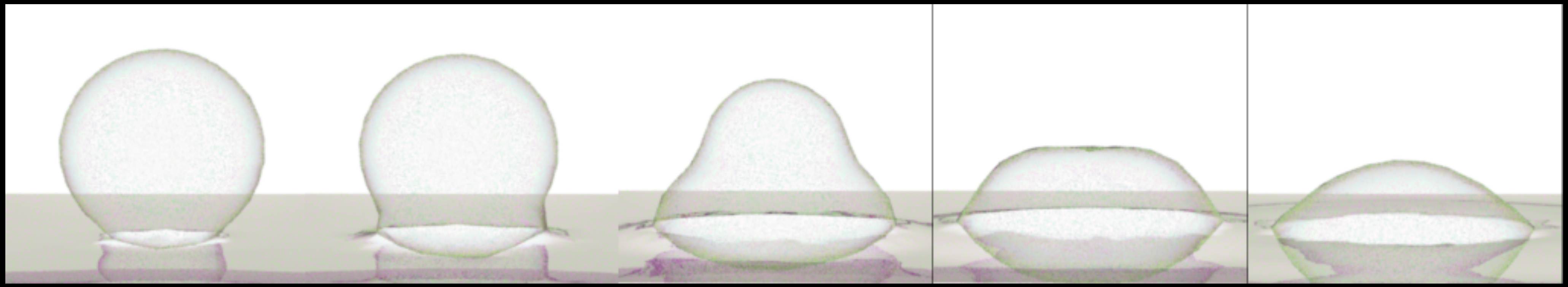



Comparision with Da et al. 2015

# Comparison to real film




experiment [Pucci et al. 2015]



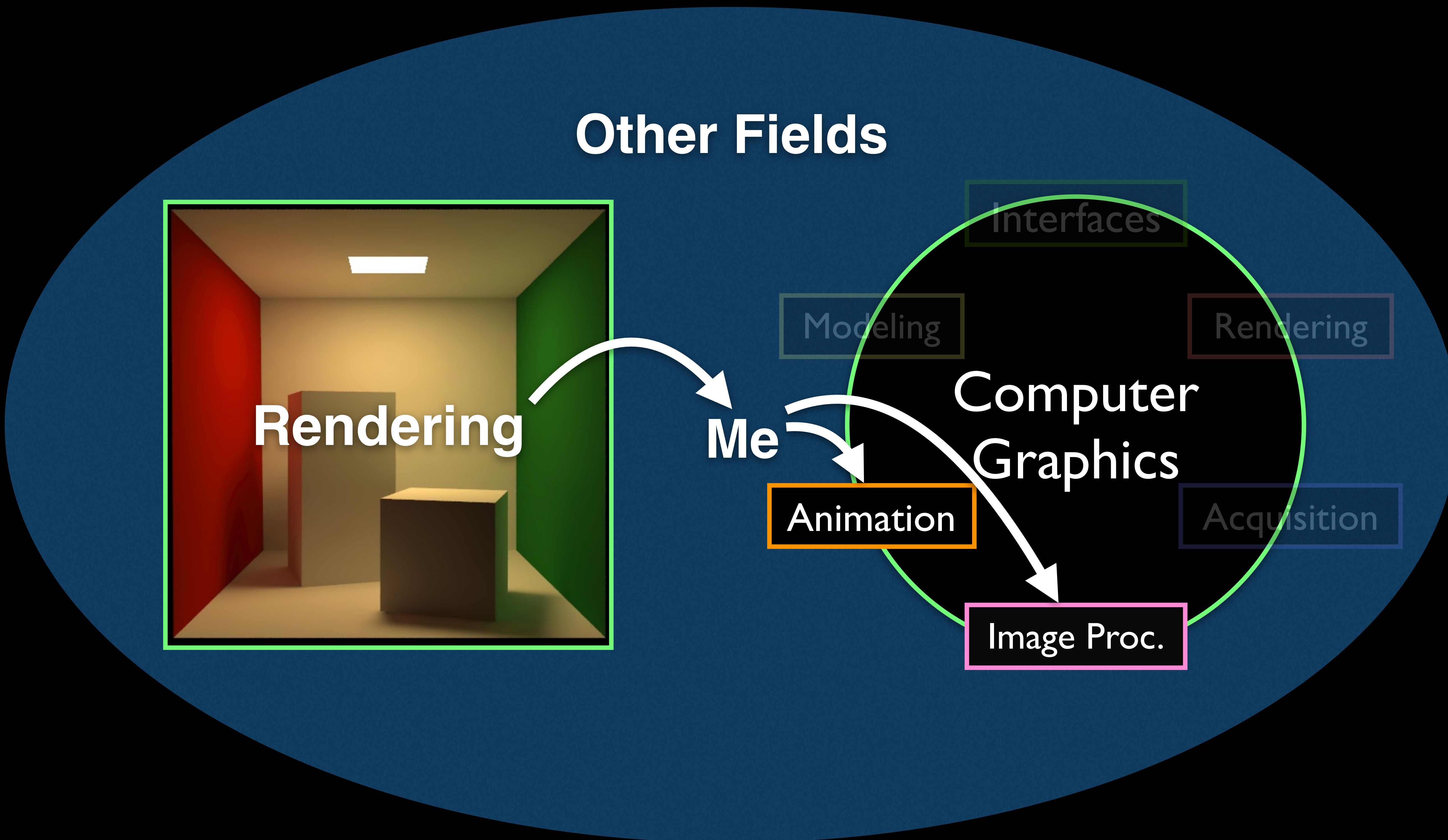
our simulation

# Summary

- Bridge physics/geometric views of film dynamics
  - Physically valid model for film animation
  - Mathematically novel geometric flow
- Very stable solver for the Plateau's problem
- Source code - https://github.com/sdsgisd/HGF

# Concluding remarks

# Thinking outside the box



# Thinking outside the box

- "Wavelet convolutional neural networks"

Image processing: Wavelet analysis of images

+

Machine learning: Convolutional neural networks

- "Hyperbolic geometric flow for soap film dynamics"

Animation: Physics simulation of soap films

+

Differential geometry: Geometric flow

# Thinking outside the box

- Interdisciplinary nature of graphics research forced me to think outside the box
- Taking insights from different fields can lead to unexpected and surprising results
- My long term goal is to make continuous effort on bridging mathematics and graphics

**Contact me if you are interested in this effort!**