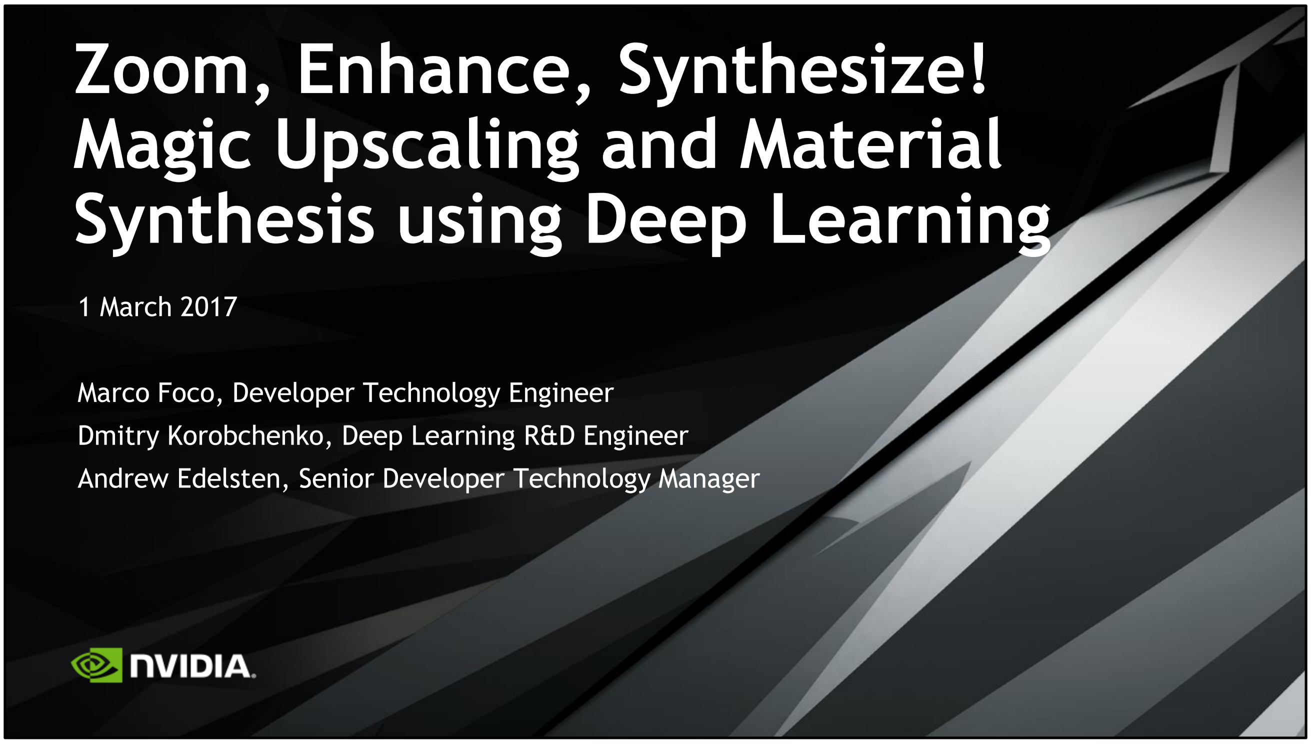


Zoom, Enhance, Synthesize! Magic Upscaling and Material Synthesis using Deep Learning

1 March 2017

Marco Foco, Developer Technology Engineer
Dmitry Korobchenko, Deep Learning R&D Engineer
Andrew Edelsten, Senior Developer Technology Manager

Session Description: Recently deep learning has revolutionized computer vision and other recognition problems. Everyday applications using such techniques are now commonplace with more advanced tasks being automated at a growing rate. During 2016, "image synthesis" techniques started to appear that used deep neural networks to apply style transfer algorithms for image restoration. The speakers review some of these techniques and demonstrate their application in image magnification to enable "super resolution" tools.

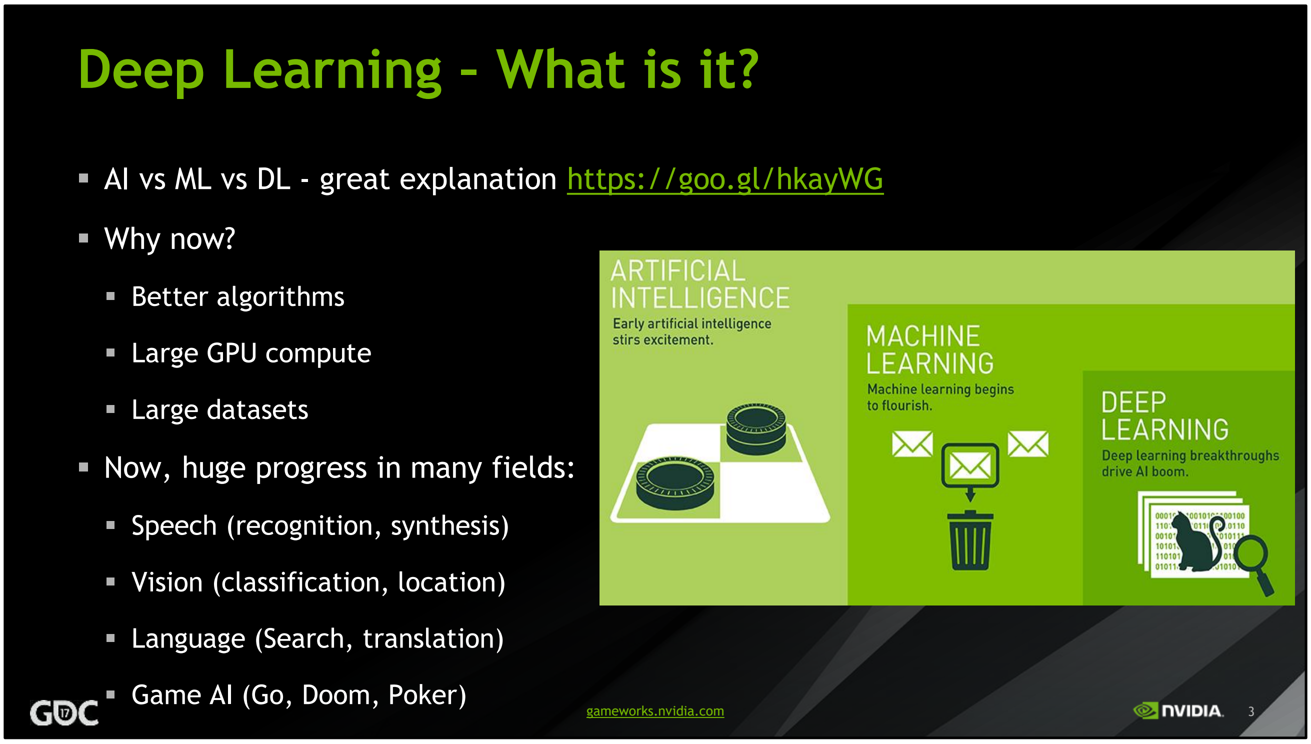
The speakers also discuss recent discoveries by NVIDIA Research that uses AI, machine learning and deep learning based approaches to greatly improve the process of creating game-ready materials. Using these novel techniques, artists can use standard DSLR, or even cell phone cameras, to create full renderable materials in minutes. The session concludes by showing how developers can integrate these methods into their existing art pipelines.

Takeaway: Attendees will gain information about the latest application of machine and deep learning for content creation and get access to new resources to improve their work.

Intended Audience: Texture artists, art directors, tool programmers, anyone interested in latest evolution of deep learning in game development.

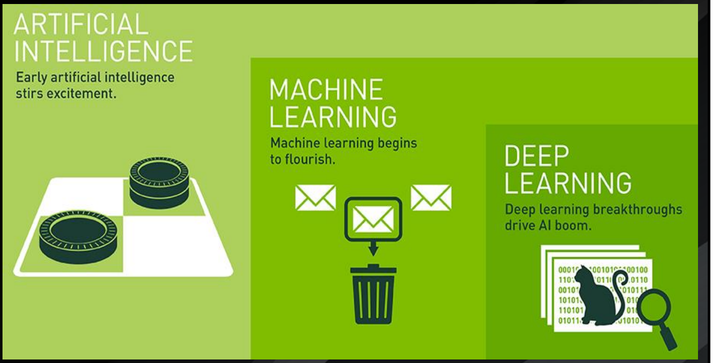
Overview

- Welcome
- What is Deep Learning?
- "GameWorks: Materials & Textures" [producers and artists rejoice]
- Examine in detail the design of one tool [coders bathe in technical details]
- Wrap up



Deep Learning - What is it?

- AI vs ML vs DL - great explanation https://goo.gl/hkayWG
- Why now?
  - Better algorithms
  - Large GPU compute
  - Large datasets
- Now, huge progress in many fields:
  - Speech (recognition, synthesis)
  - Vision (classification, location)
  - Language (Search, translation)
  - Game AI (Go, Doom, Poker)


ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.
MACHINE LEARNING
Machine learning begins to flourish.
DEEP LEARNING
Deep learning breakthroughs drive AI boom.

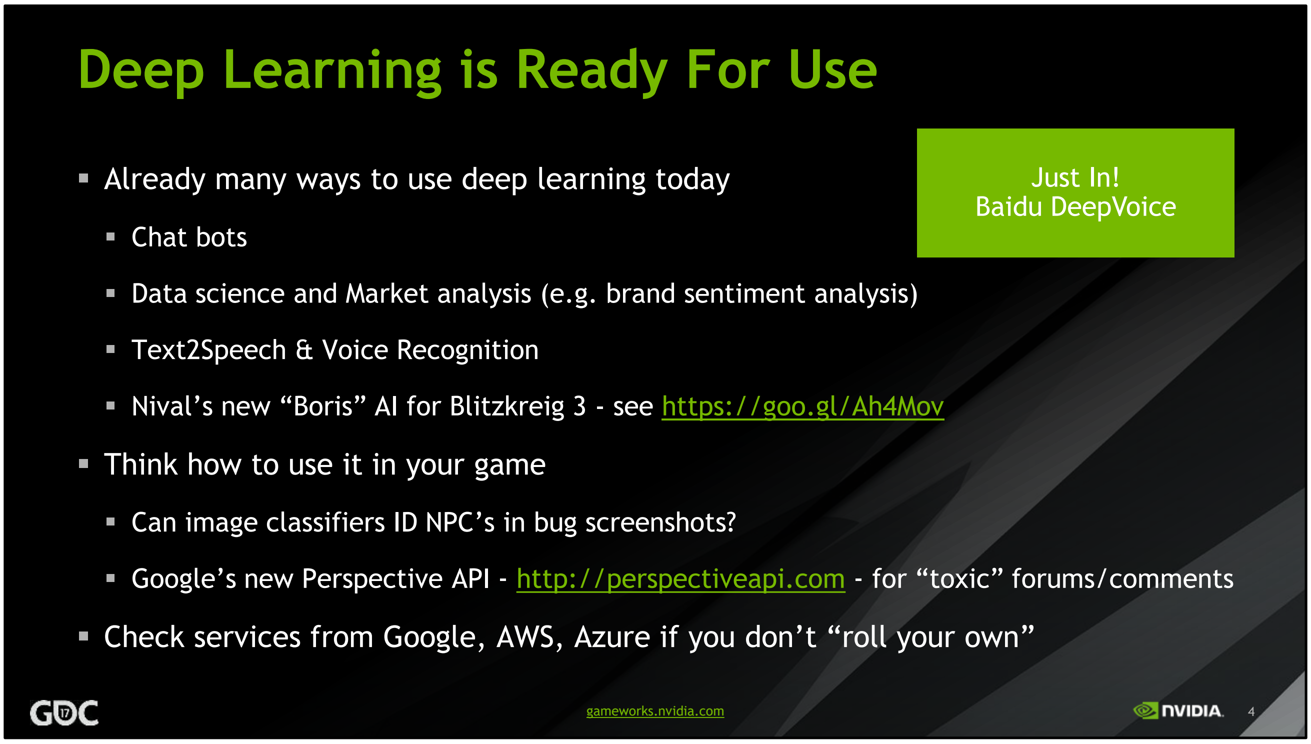
GDC gameworks.nvidia.com NVIDIA 3

Machine Learning at its most basic is the practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something in the world. So rather than hand-coding software routines with a specific set of instructions to accomplish a particular task, the machine is “trained” using large amounts of data and algorithms that give it the ability to learn how to perform the task.

One approach to ML was “artificial neural networks” – basically use “simple” math in a distributed way to try and mimic the way we think neurons in the brain work. Anyway, for years ANN resulted in nothing until:
Prof Hinton @ Uni of Toronto made the algorithms parallel, and then the algorithms were put on GPU. Then training sets exploded.

Using DL everyday.. A lot!
Web search
Siri/Google Now
Facebook image/face tagging
Language translation
Style transfer

Neural networks are so useful why now?
- Better algorithms – academics never stopped researching.. They just couldn’t try out til recently (eg RNN LSTM invented in 1997 -- Hochreiter, Sepp; and Schmidhuber, Jürgen; Long Short-Term Memory, Neural Computation, 9(8):1735–1780, 1997)
- Large datasets – the digital lifestyles we live, leads to huge data collection
- Large compute – turns out, the math for NN is HIGHLY parallel.. just like graphics! Yay GPU!



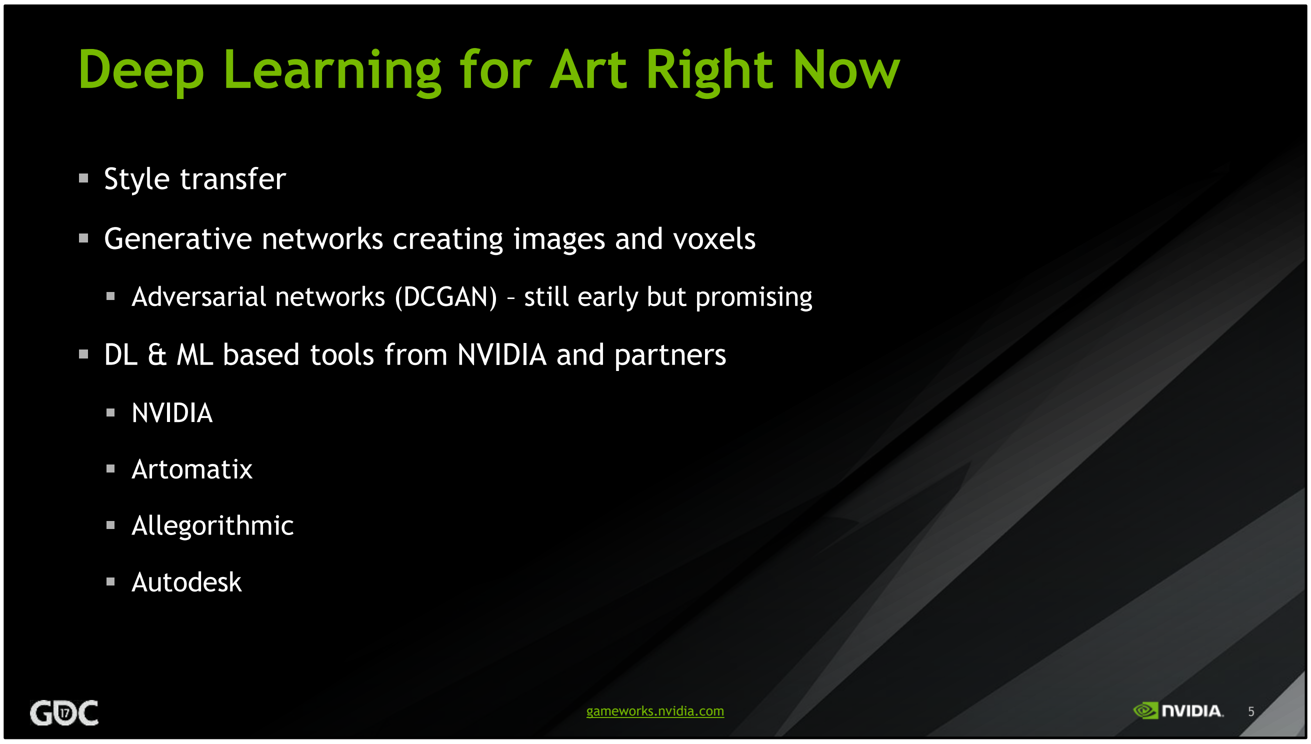
Deep Learning is Ready For Use

Just In!
Baidu DeepVoice

- Already many ways to use deep learning today
  - Chat bots
  - Data science and Market analysis (e.g. brand sentiment analysis)
  - Text2Speech & Voice Recognition
  - Nival's new "Boris" AI for Blitzkreig 3 - see https://goo.gl/Ah4Mov
- Think how to use it in your game
  - Can image classifiers ID NPC's in bug screenshots?
  - Google's new Perspective API - http://perspectiveapi.com - for "toxic" forums/comments
- Check services from Google, AWS, Azure if you don't "roll your own"

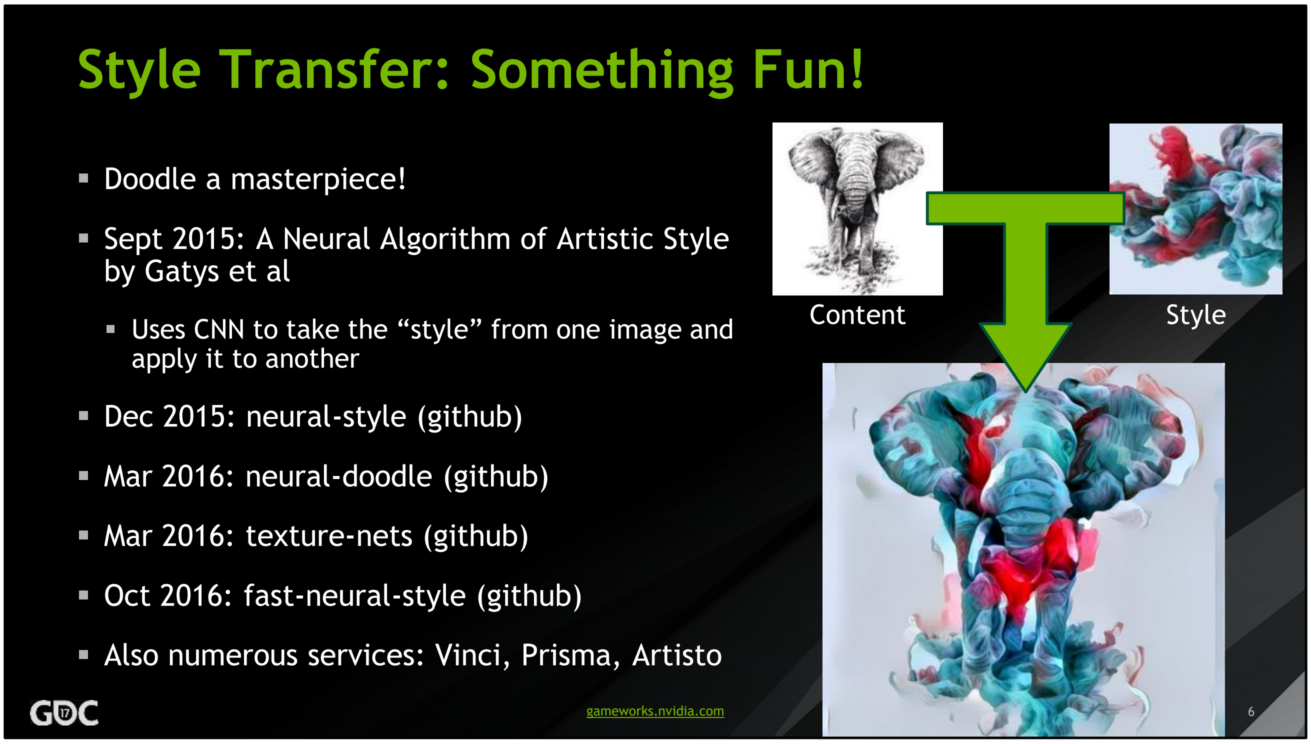
gameworks.nvidia.com

4



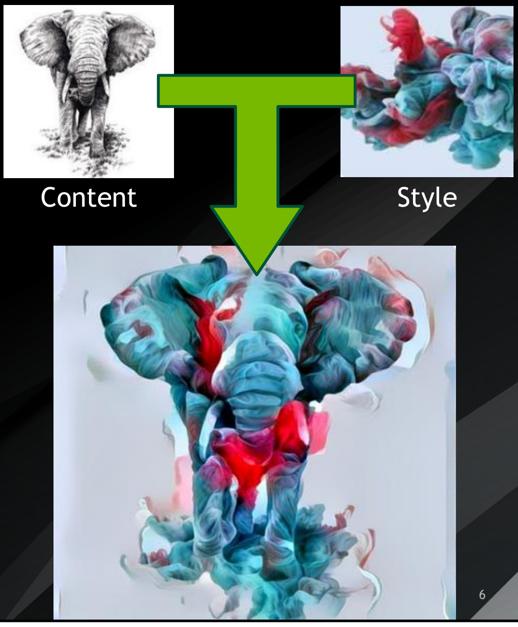
Deep Learning for Art Right Now

- Style transfer
- Generative networks creating images and voxels
  - Adversarial networks (DCGAN) – still early but promising
- DL & ML based tools from NVIDIA and partners
  - NVIDIA
  - Artomatix
  - Allegorithmic
  - Autodesk



Style Transfer: Something Fun!

- Doodle a masterpiece!
- Sept 2015: A Neural Algorithm of Artistic Style by Gatys et al
  - Uses CNN to take the "style" from one image and apply it to another
- Dec 2015: neural-style (github)
- Mar 2016: neural-doodle (github)
- Mar 2016: texture-nets (github)
- Oct 2016: fast-neural-style (github)
- Also numerous services: Vinci, Prisma, Artisto



gameworks.nvidia.com

6

References:

A Neural Algorithm of Artistic Style paper by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge

https://github.com/jcjohnson/neural-style - github repo by Justin Johnson

https://github.com/jcjohnson/fast-neural-style – github repo by Justin Johnson

https://github.com/alexjc/neural-doodle - github repo by @alexjc

Services:

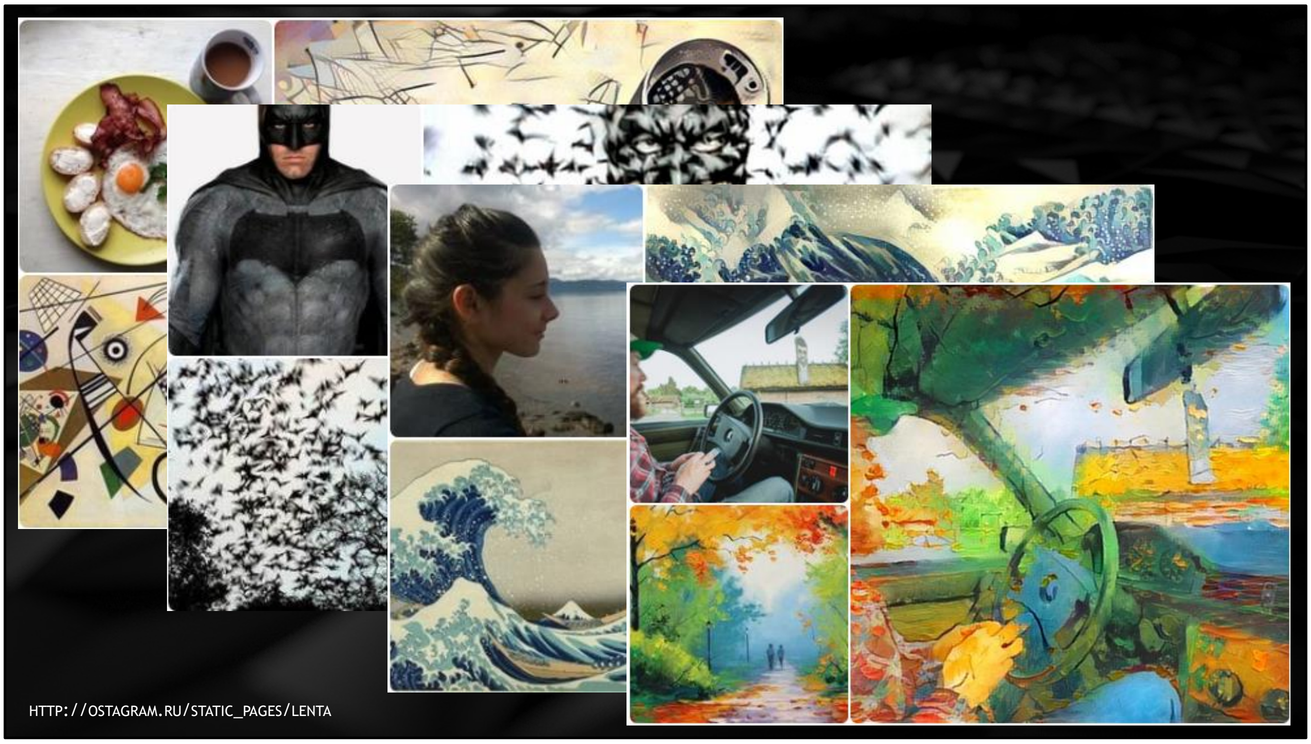
http://ostagram.ru/static_pages/lenta

https://www.instapainting.com/ai-painter

iOS app (calls out to server) http://prisma-ai.com/

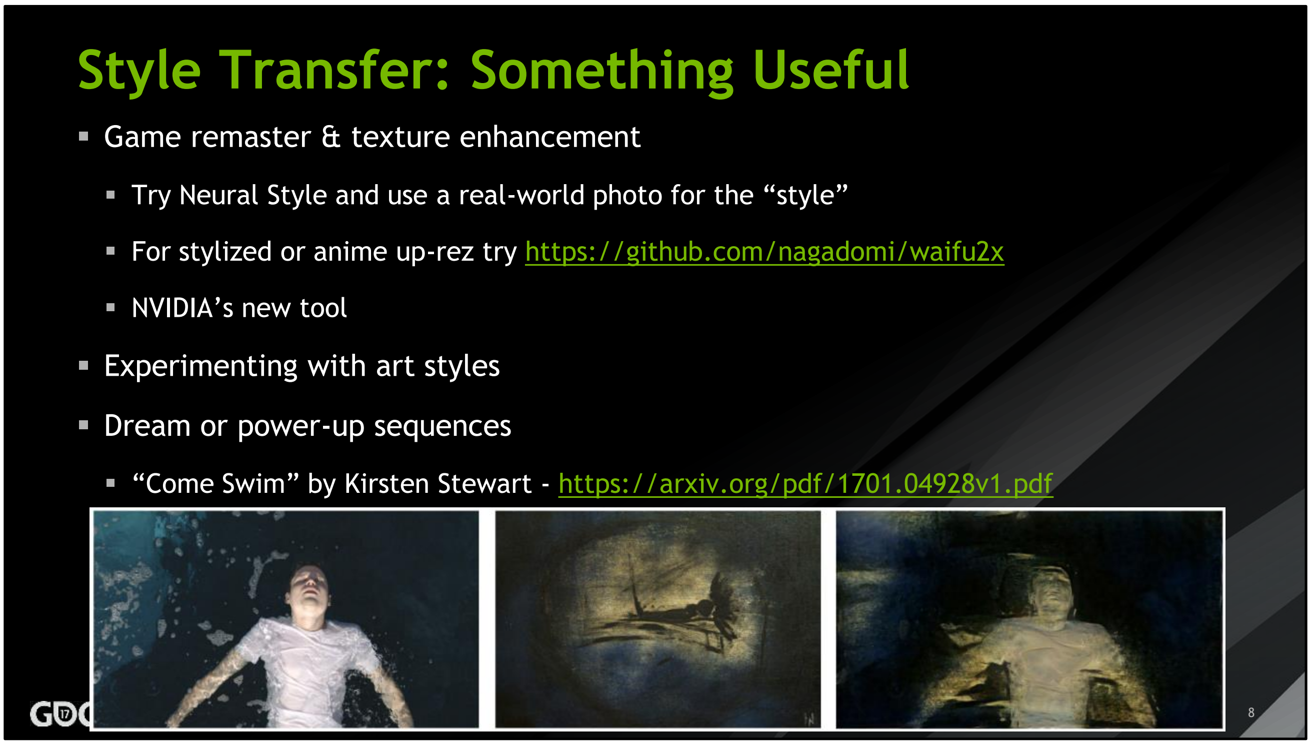
Run your own web service: https://github.com/hletrd/neural_style

Decent tutorial: http://www.makeuseof.com/tag/create-neural-paintings-deepstyle-ubuntu/



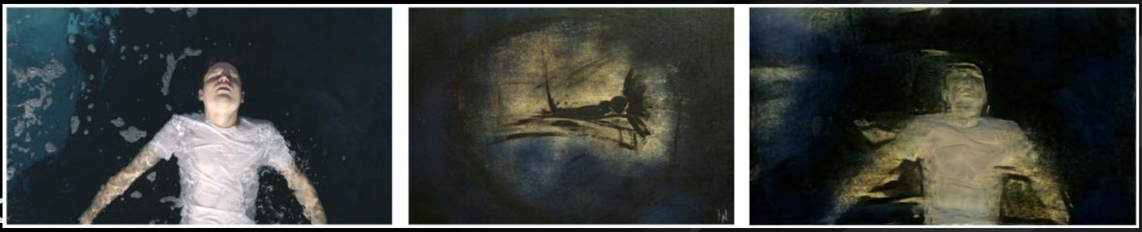
Can generate some pretty amazing artwork very easily.

But in addition to being a great toy, there is great potential – I mean, the AI is actually drawing pixels in a meaningful way.



Style Transfer: Something Useful

- Game remaster & texture enhancement
  - Try Neural Style and use a real-world photo for the “style”
  - For stylized or anime up-rez try https://github.com/nagadomi/waifu2x
  - NVIDIA’s new tool
- Experimenting with art styles
- Dream or power-up sequences
  - “Come Swim” by Kirsten Stewart - https://arxiv.org/pdf/1701.04928v1.pdf



Come Swim paper - https://arxiv.org/pdf/1701.04928v1.pdf

Bhautik J Joshi - Research Engineer, Adobe

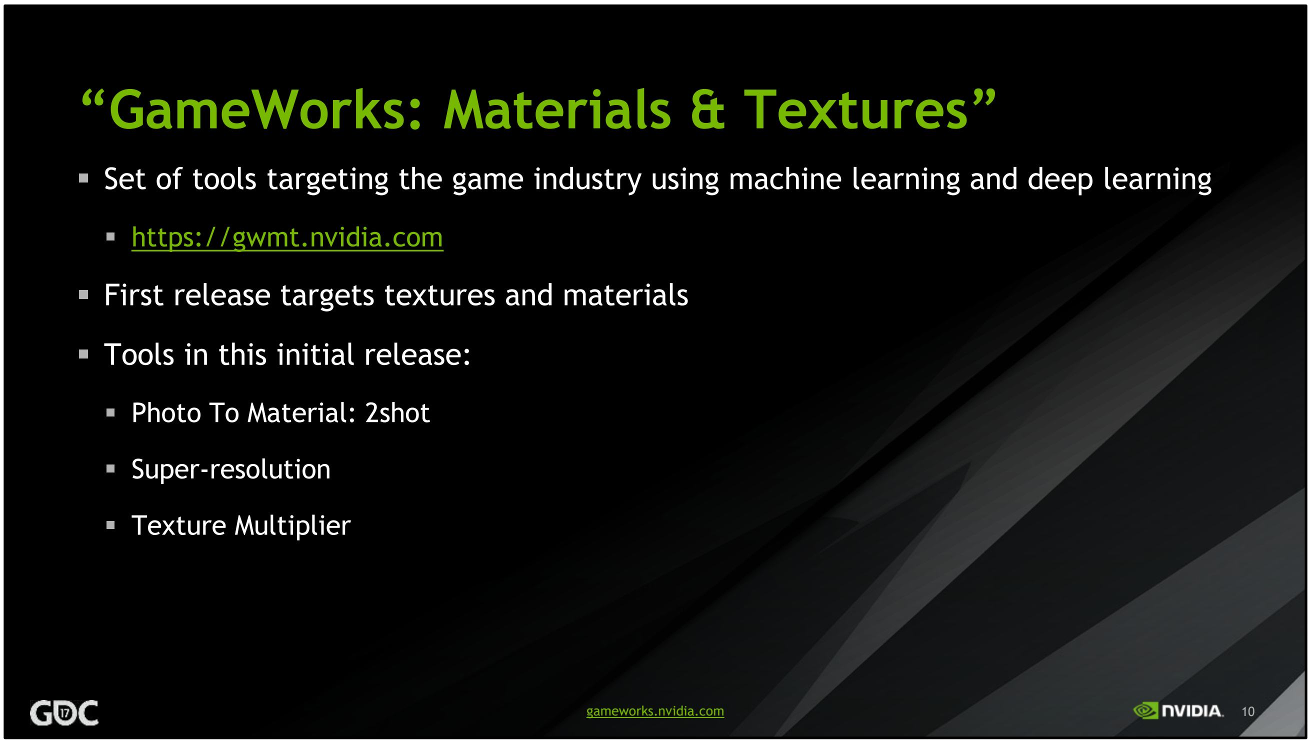
Kristen Stewart - Director, Come Swim

David Shapiro - Producer, Starlight Studios

https://www.theguardian.com/film/2017/jan/20/kristen-stewart-research-paper-neural-style-transfer

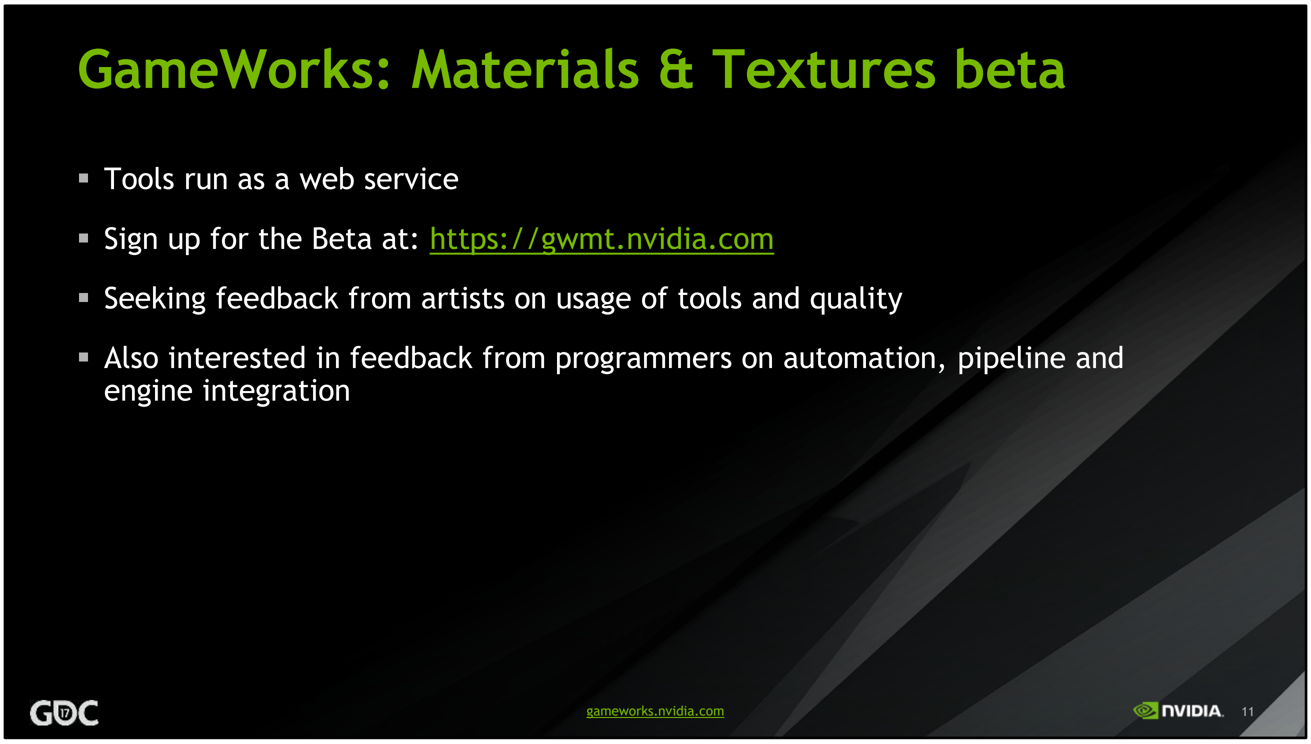
NVIDIA's Goals for DL in Game Development

- Looking at all the research, clearly there's scope for tools based on DL
- Goals:
  - Expand the use of deep learning into content creation
  - Remove the mundane and repetitive
  - Promote increased creativity, realism and experimentation



“GameWorks: Materials & Textures”

- Set of tools targeting the game industry using machine learning and deep learning
  - https://gwmt.nvidia.com
- First release targets textures and materials
- Tools in this initial release:
  - Photo To Material: 2shot
  - Super-resolution
  - Texture Multiplier



GameWorks: Materials & Textures beta

- Tools run as a web service
- Sign up for the Beta at: https://gwmt.nvidia.com
- Seeking feedback from artists on usage of tools and quality
- Also interested in feedback from programmers on automation, pipeline and engine integration

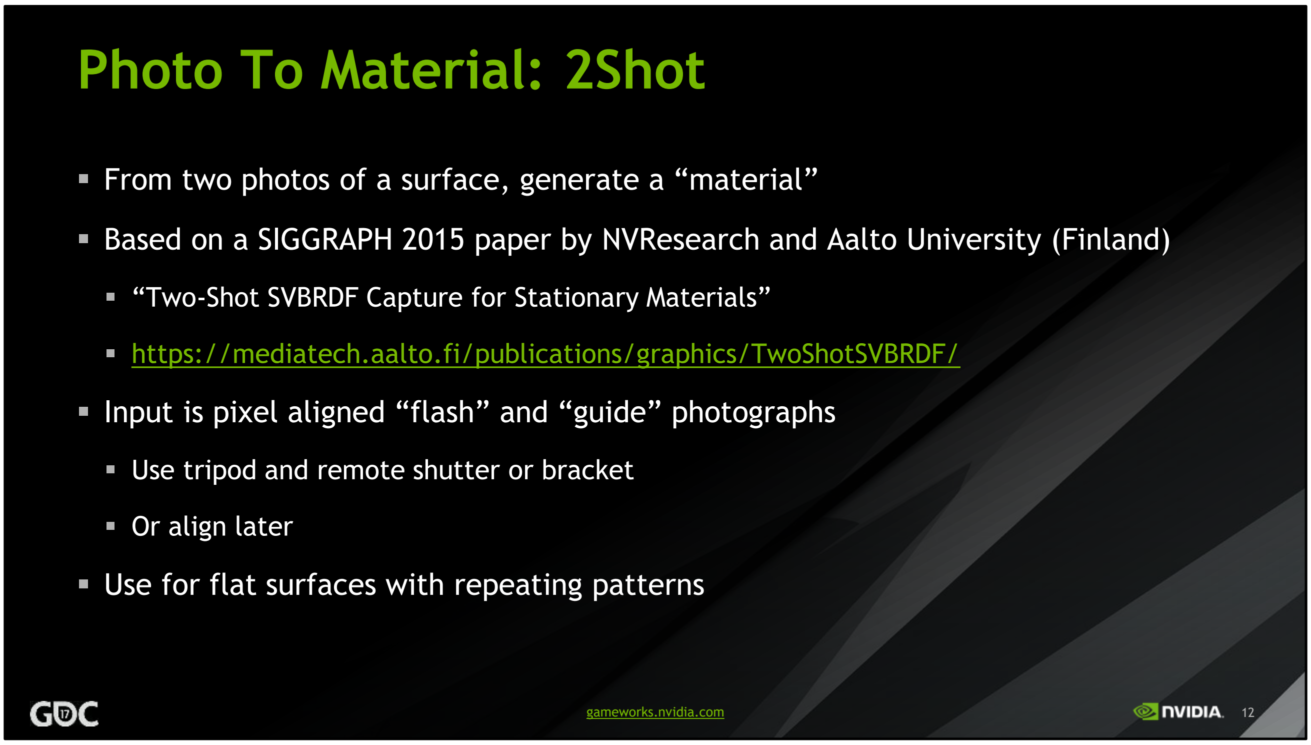
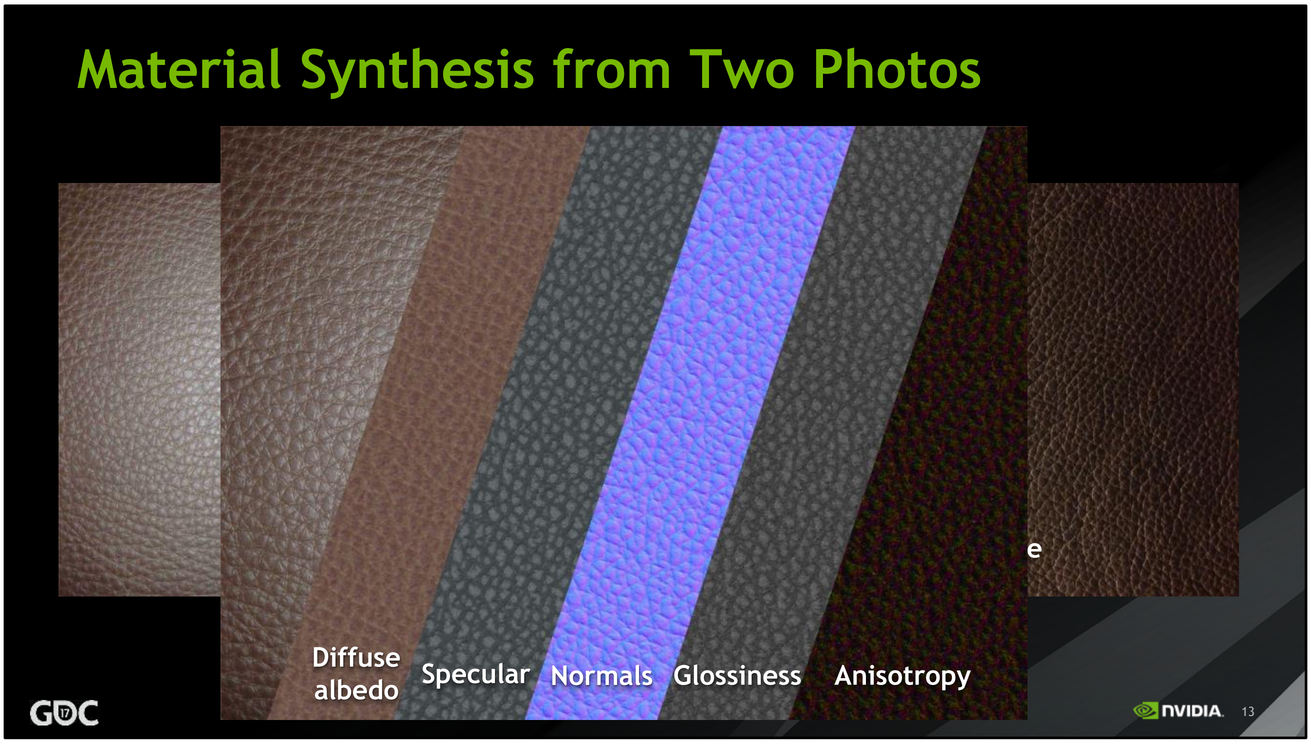
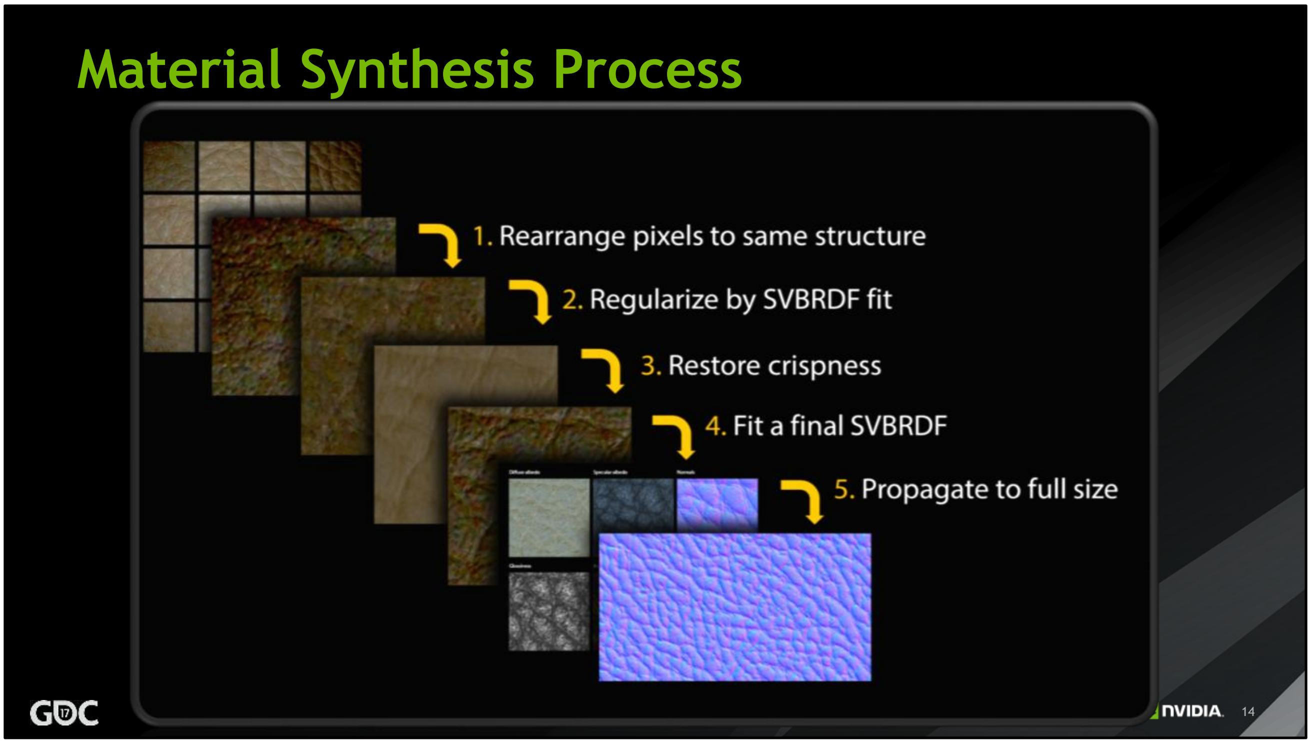


Photo To Material: 2Shot

- From two photos of a surface, generate a "material"
- Based on a SIGGRAPH 2015 paper by NVResearch and Aalto University (Finland)
  - "Two-Shot SVBRDF Capture for Stationary Materials"
  - https://mediatech.aalto.fi/publications/graphics/TwoShotSVBRDF/
- Input is pixel aligned "flash" and "guide" photographs
  - Use tripod and remote shutter or bracket
  - Or align later
- Use for flat surfaces with repeating patterns



Material Synthesis from Two Photos



Material Synthesis Process

1. Rearrange pixels to same structure
2. Regularize by SVBRDF fit
3. Restore crispness
4. Fit a final SVBRDF
5. Propagate to full size

14

SVBRDF – spatially varying bidirectional reflectance distribution function

Demo

Photo To Material: 2Shot

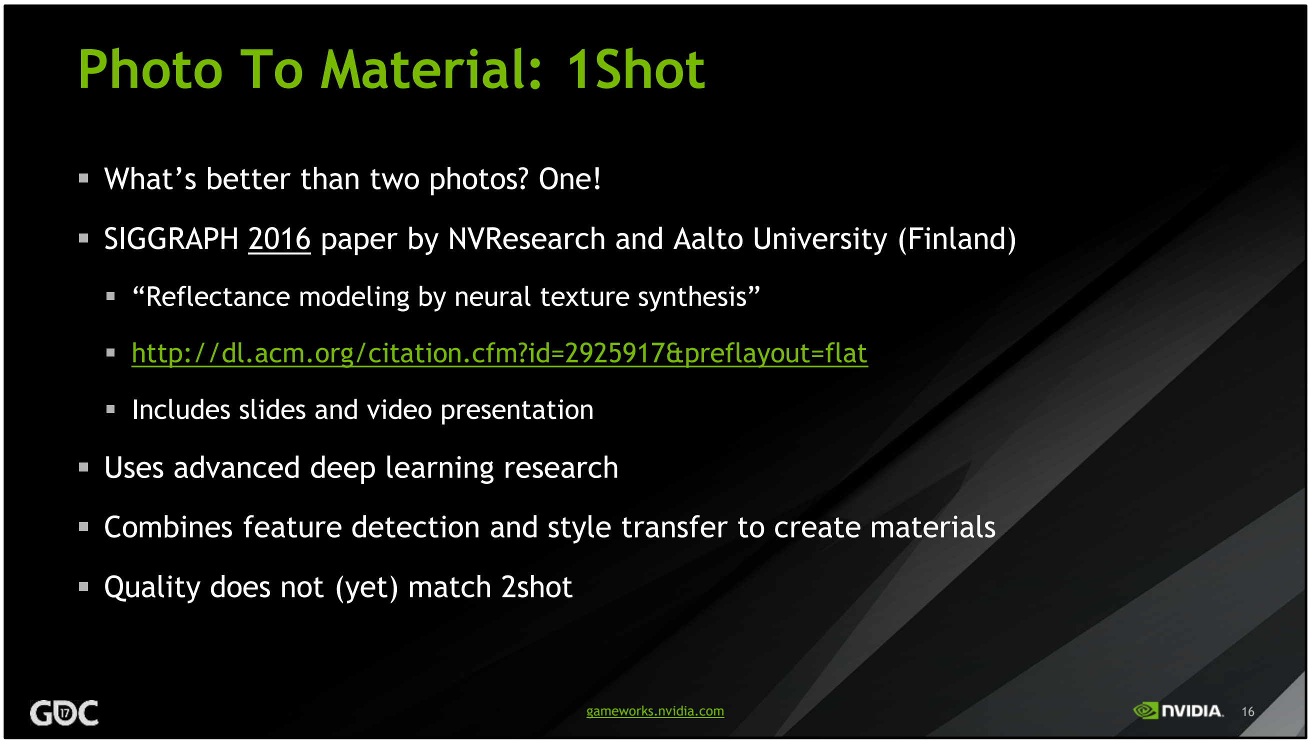
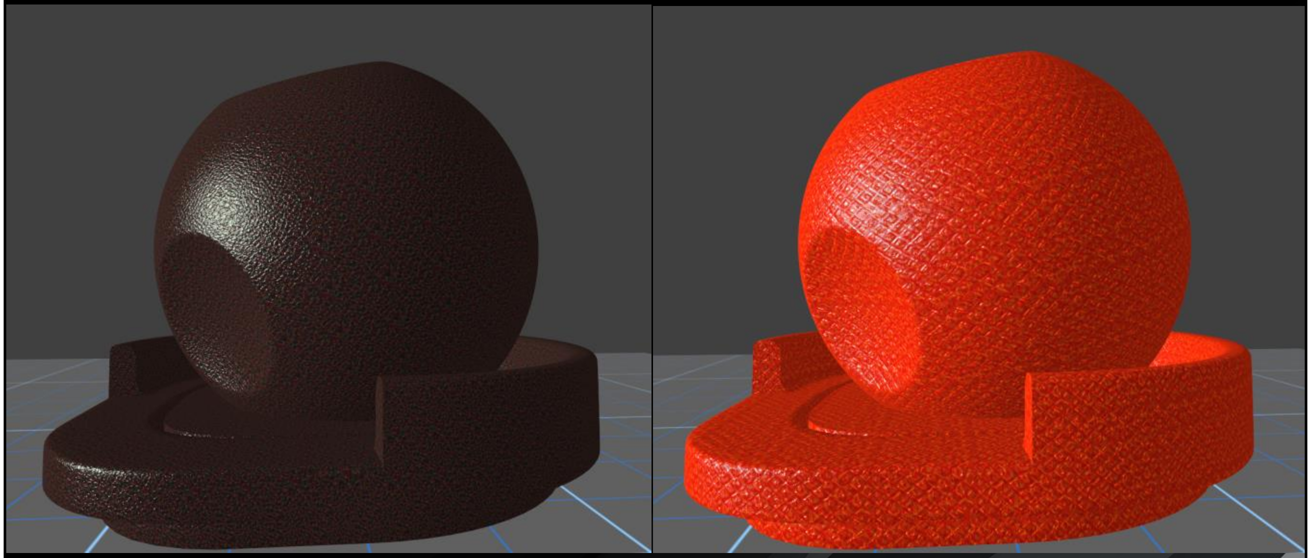


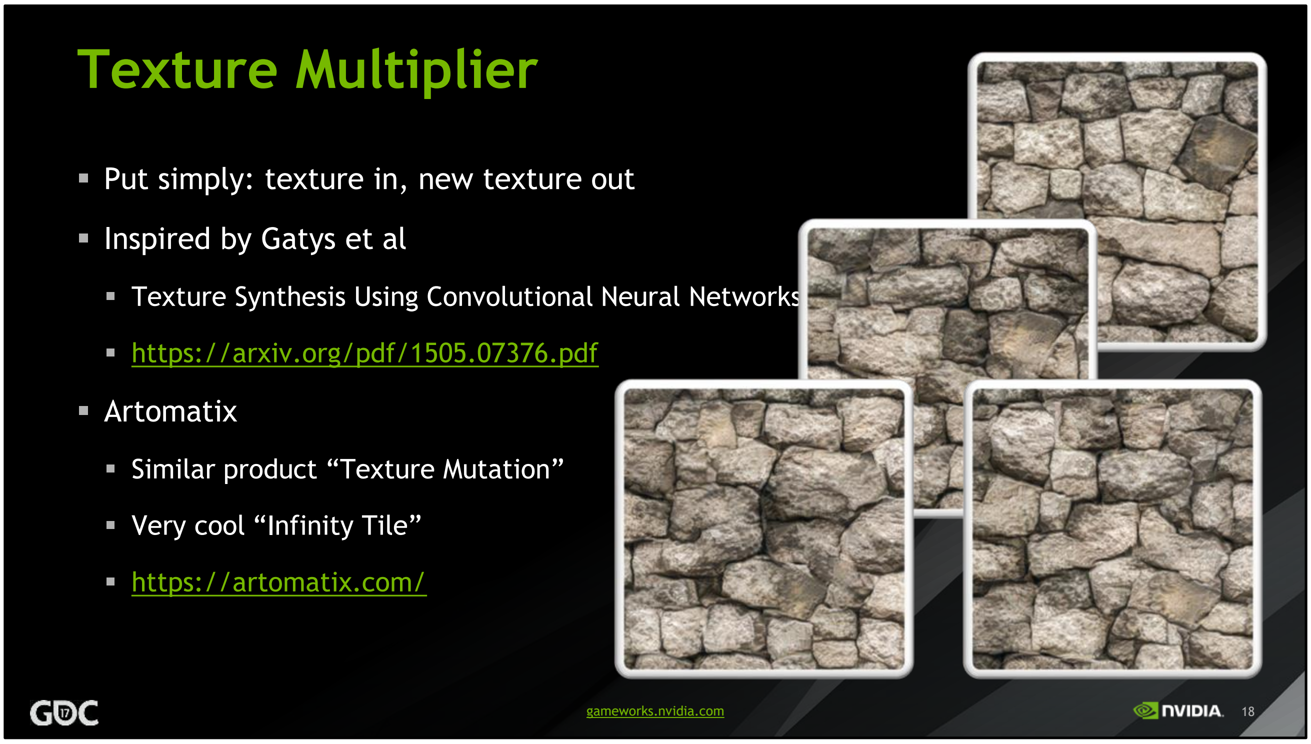
Photo To Material: 1Shot

- What's better than two photos? One!
- SIGGRAPH 2016 paper by NVResearch and Aalto University (Finland)
  - "Reflectance modeling by neural texture synthesis"
  - http://dl.acm.org/citation.cfm?id=2925917&preflayout=flat
  - Includes slides and video presentation
- Uses advanced deep learning research
- Combines feature detection and style transfer to create materials
- Quality does not (yet) match 2shot

gameworks.nvidia.com 16

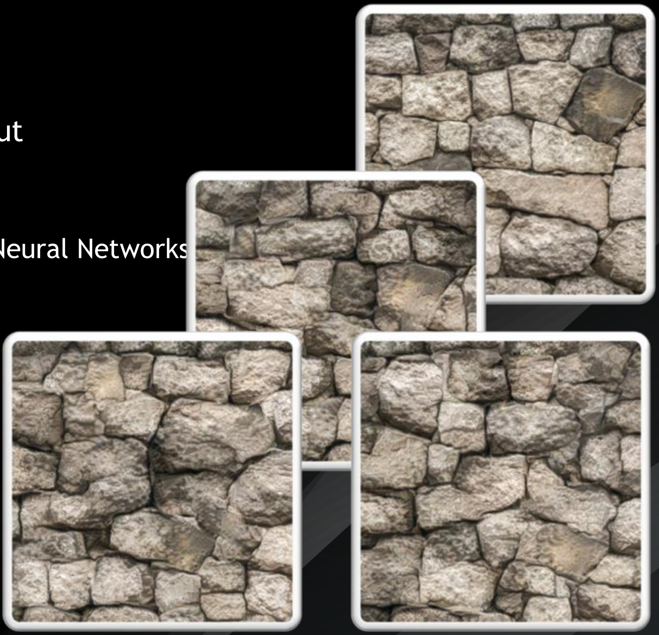
1shot - EARLY Previews





Texture Multiplier

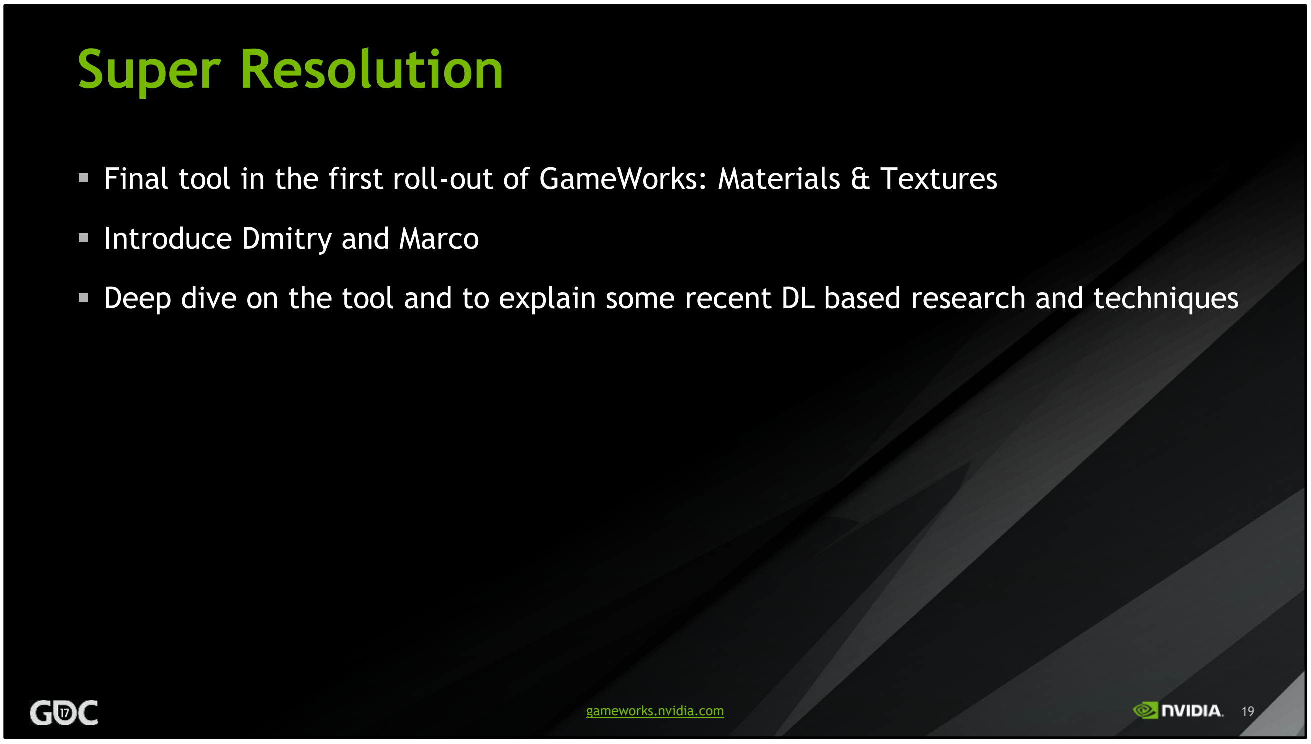
- Put simply: texture in, new texture out
- Inspired by Gatys et al
  - Texture Synthesis Using Convolutional Neural Networks
  - https://arxiv.org/pdf/1505.07376.pdf
- Artomatix
  - Similar product "Texture Mutation"
  - Very cool "Infinity Tile"
  - https://artomatix.com/



GDC 17 gameworks.nvidia.com NVIDIA 18

Currently "Beta"

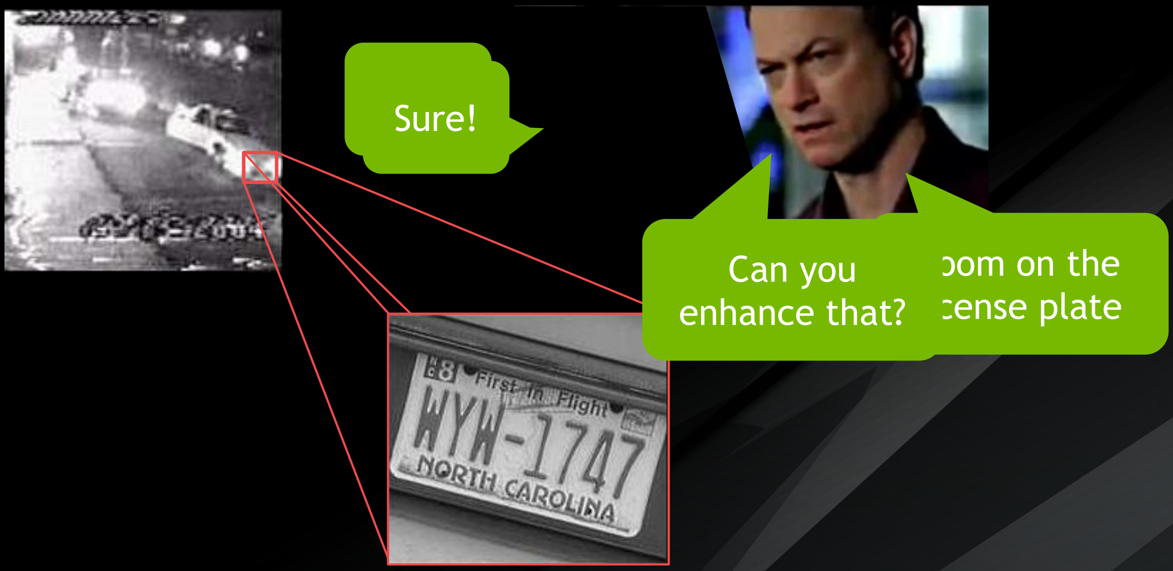
Some artifacts – 256x256 now, with 512 and 1024 coming

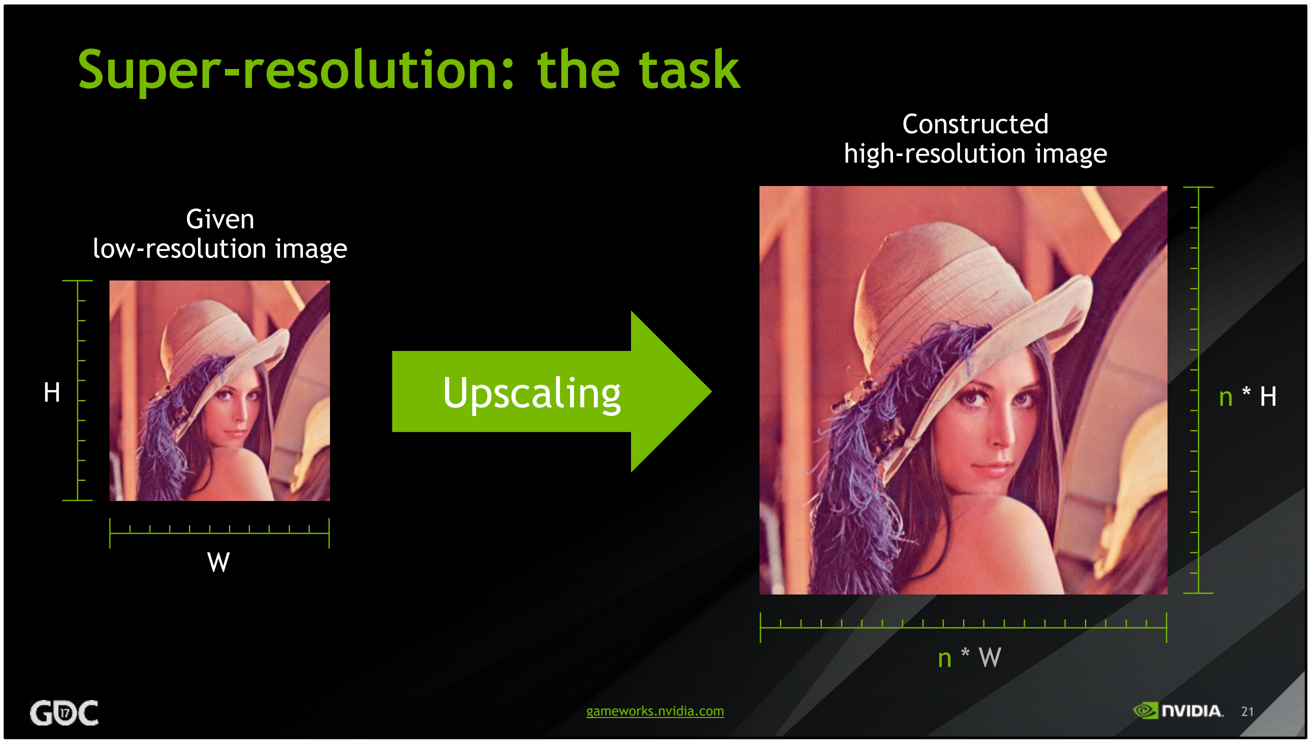


Super Resolution

- Final tool in the first roll-out of GameWorks: Materials & Textures
- Introduce Dmitry and Marco
- Deep dive on the tool and to explain some recent DL based research and techniques

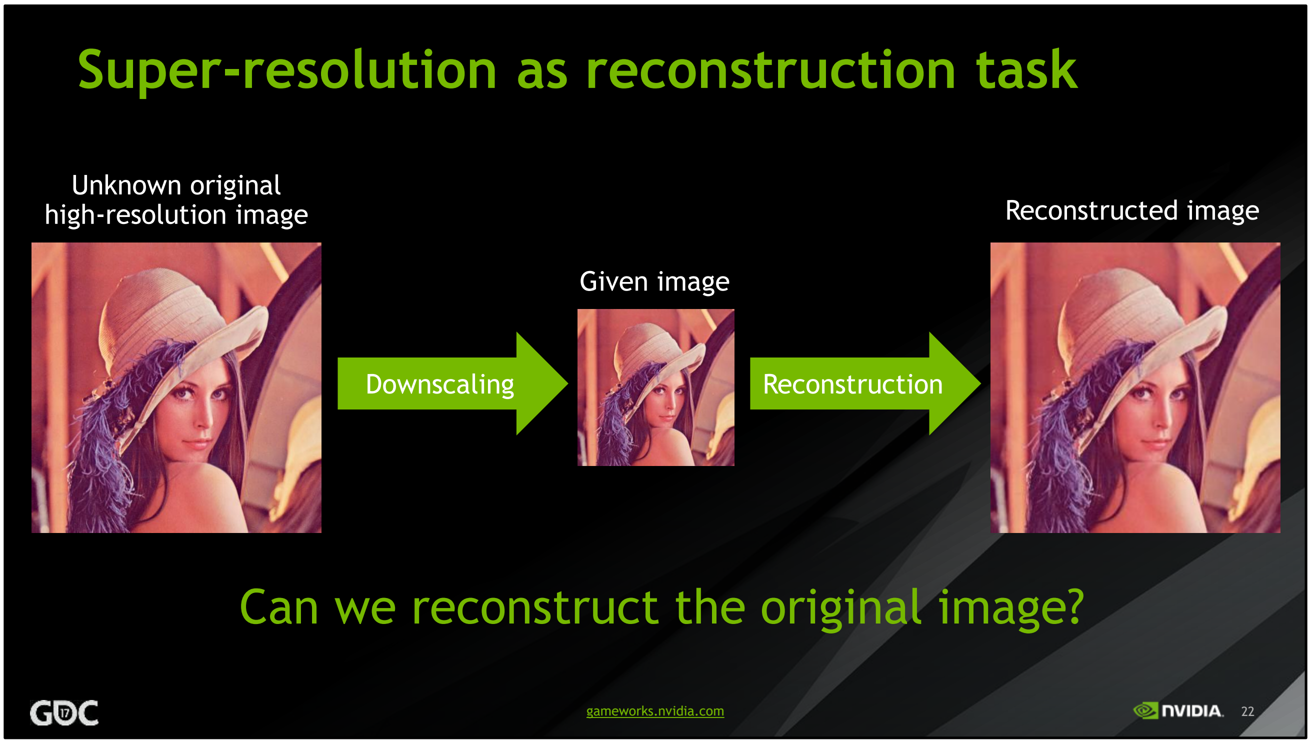
Zoom! Enhance!



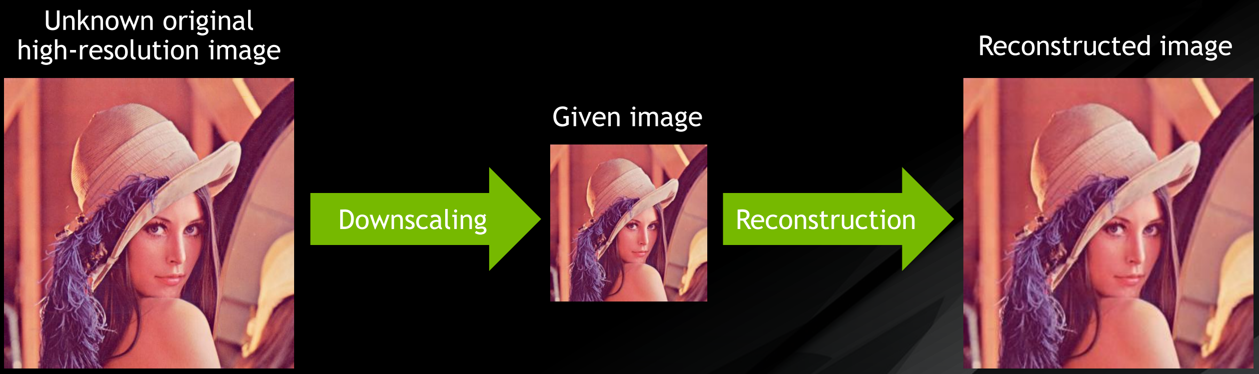


Super-resolution: the task

The task is to “generate” a bigger image from a smaller one. If we want to use machine learning to do this, we can create two set, one of big images and one of their downscaled version, and train our system with these two sets

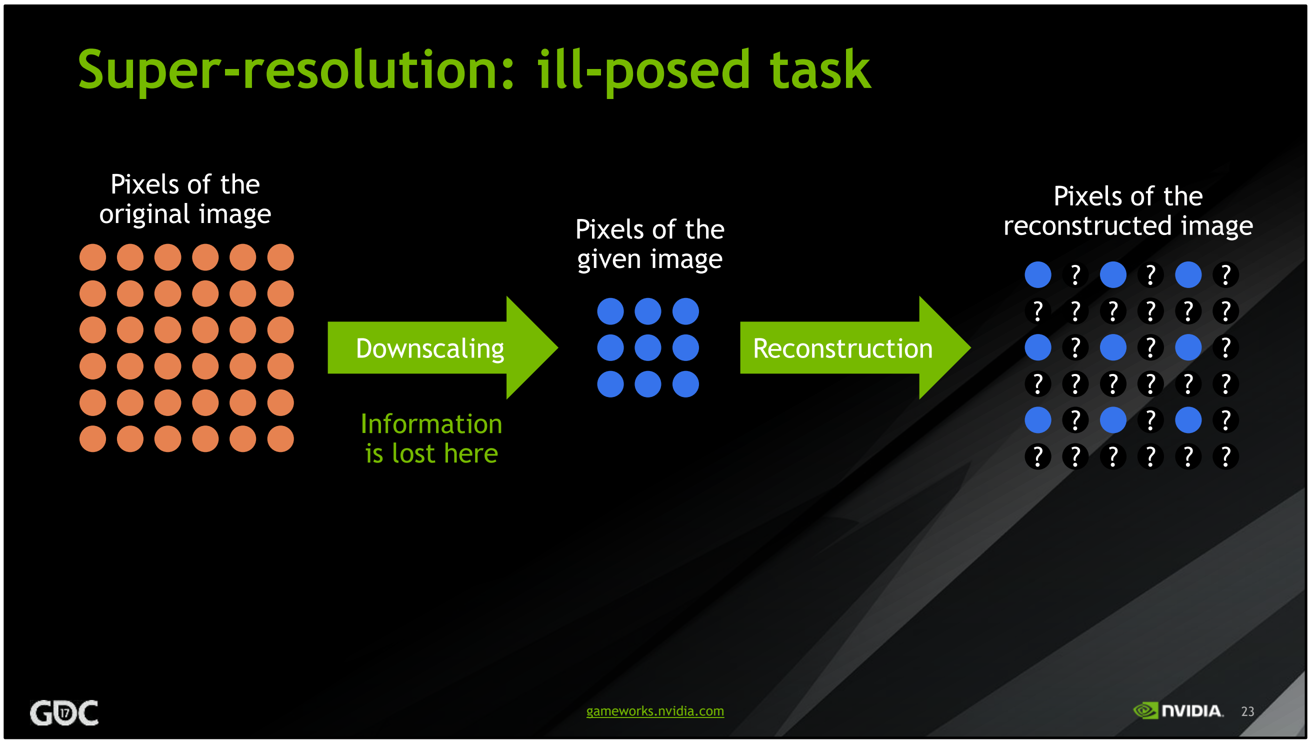


Super-resolution as reconstruction task



Can we reconstruct the original image?

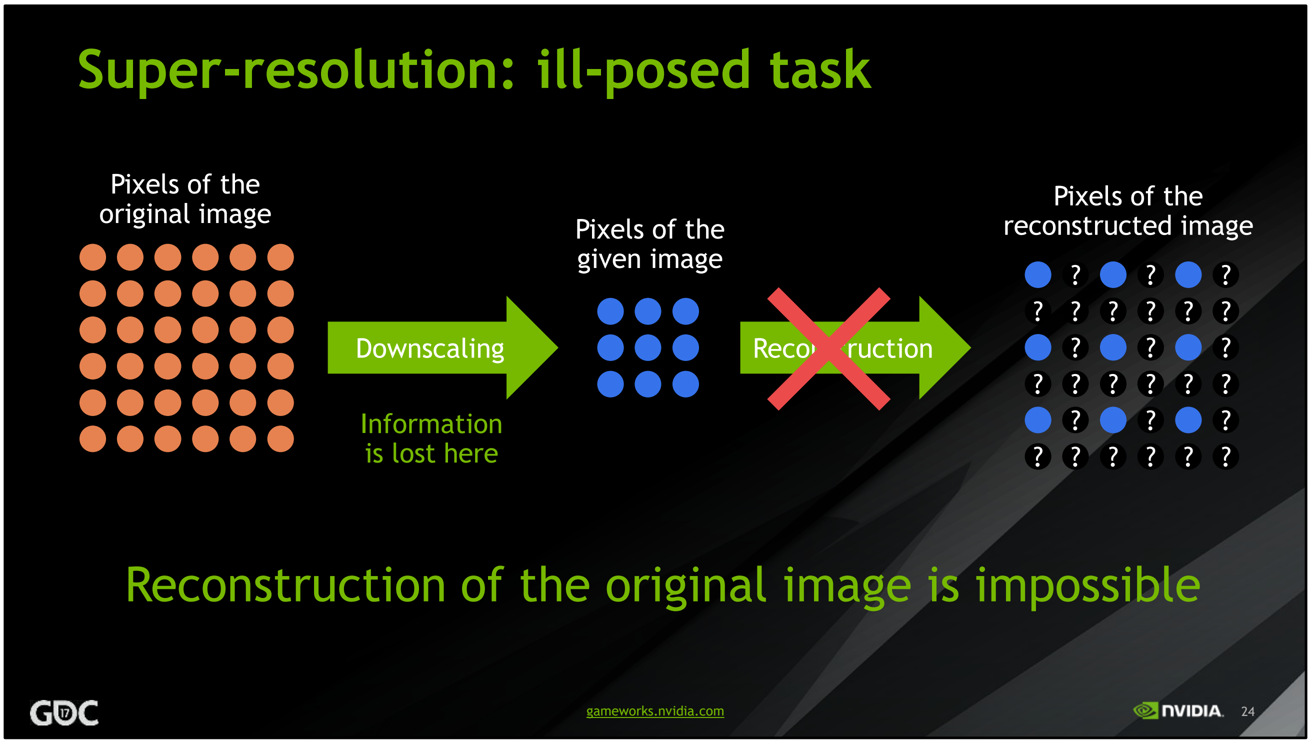
Another option is to see our task as a reconstruction. If we make the downscaling part of the process, we can use just one set, and the expected value for our system will be the input itself



Super-resolution: ill-posed task

Pixels of the original image
Downscaling
Information is lost here
Pixels of the given image
Reconstruction
Pixels of the reconstructed image
?

But the problem is ill-posed. We first remove some information, and then try to reconstruct the image using less data (1/4, in this case, 1/n^2 for a downscale factor n)



Super-resolution: ill-posed task

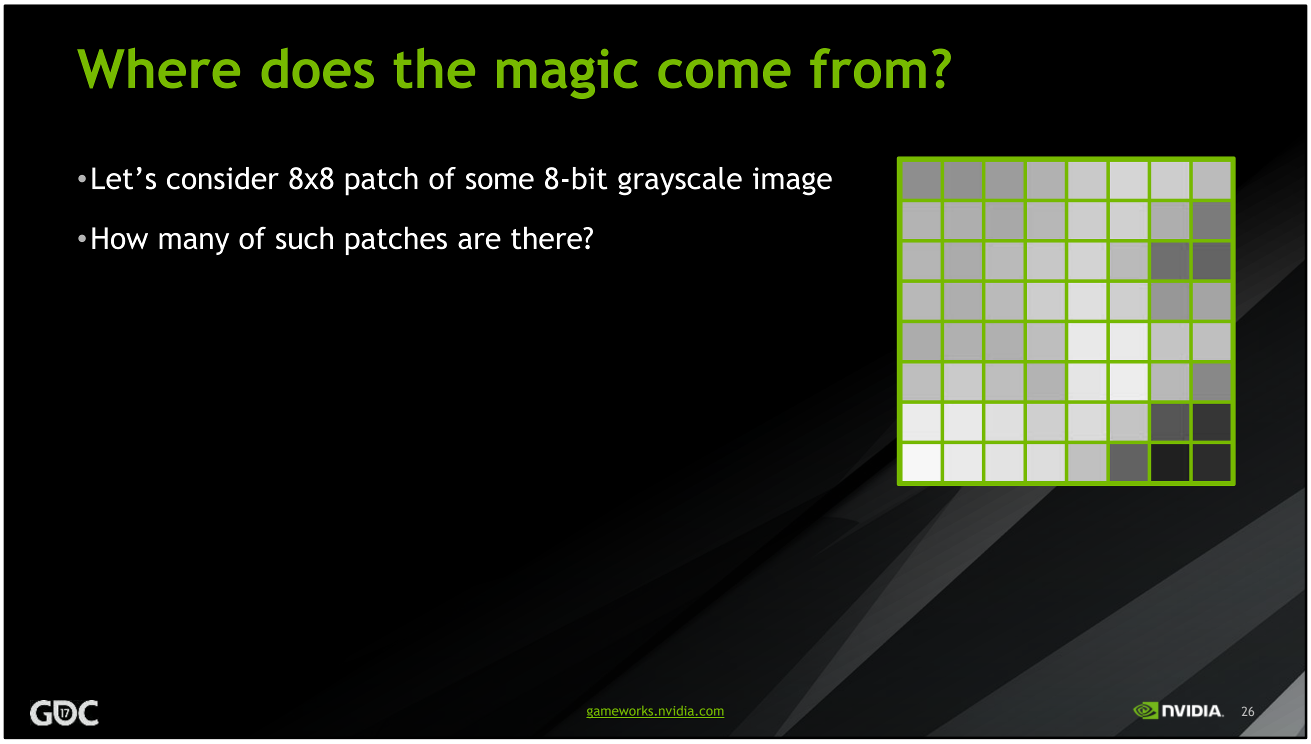
Pixels of the original image
Downscaling
Information is lost here
Pixels of the given image
Reconstruction
Pixels of the reconstructed image
?

Reconstruction of the original image is impossible

So we can’t reconstruct the original image, the information is missing!

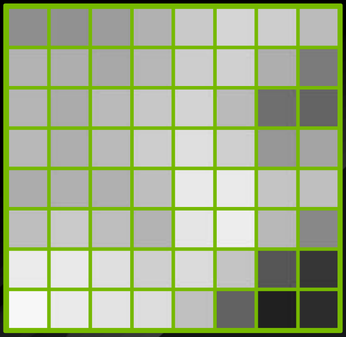
Super-resolution: ill-posed task

OR DO YOU?



Where does the magic come from?

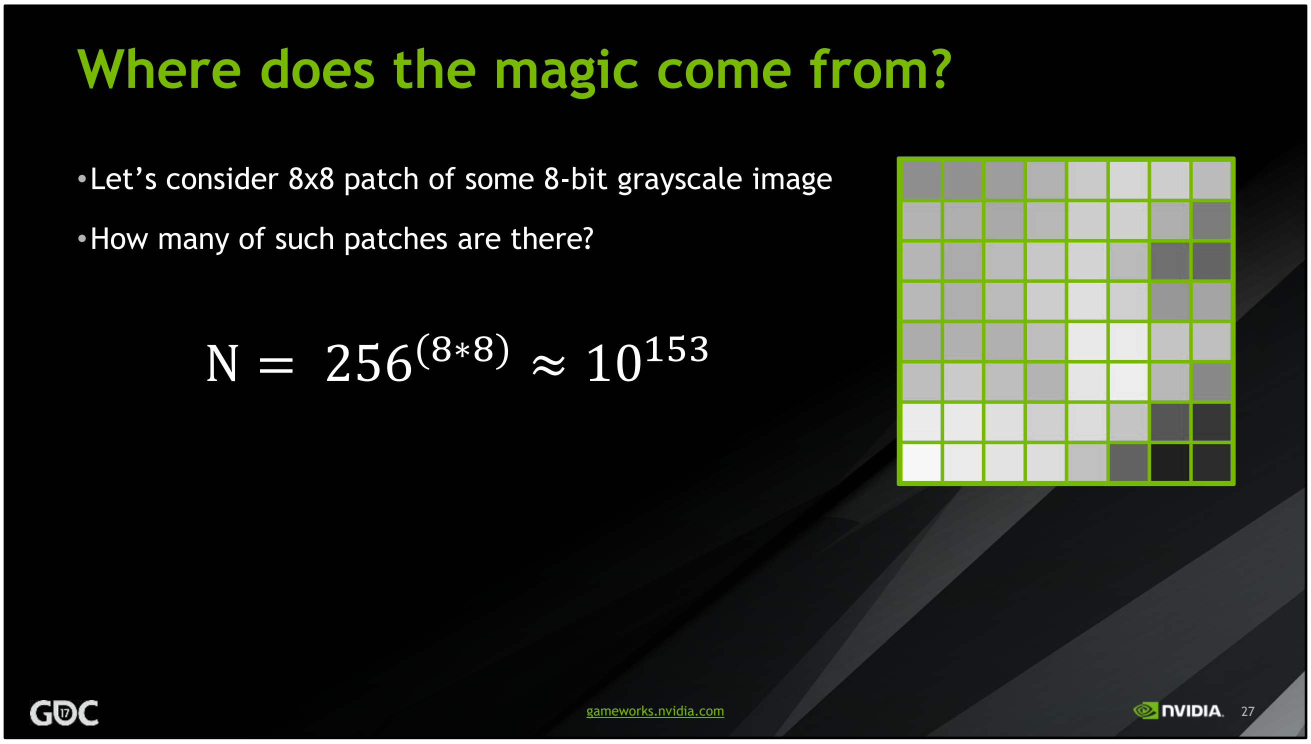
- Let's consider 8x8 patch of some 8-bit grayscale image
- How many of such patches are there?



gameworks.nvidia.com

26

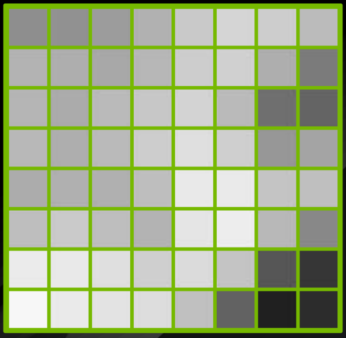
Let's consider a small portion of the original image, say 8x8 patch, and let's consider a single channel of 8 bit.



Where does the magic come from?

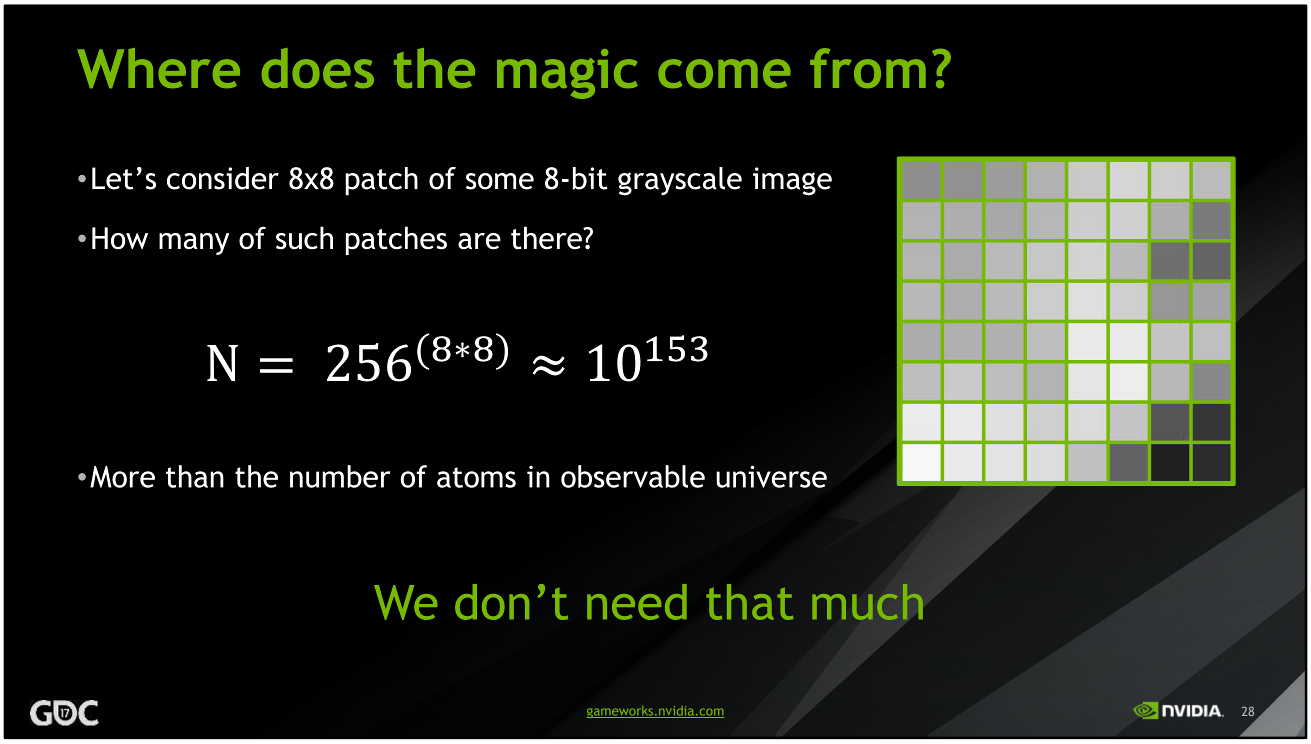
- Let’s consider 8x8 patch of some 8-bit grayscale image
- How many of such patches are there?

$$N = 256^{(8*8)} \approx 10^{153}$$



gameworks.nvidia.com 27

The number of possible values for the pixel is 256, and the number of pixels is 8x8=64, so the total number of possible images is quite big



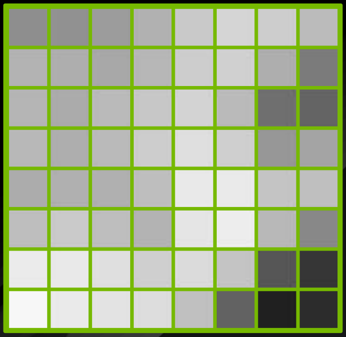
Where does the magic come from?

- Let's consider 8x8 patch of some 8-bit grayscale image
- How many of such patches are there?

$$N = 256^{(8*8)} \approx 10^{153}$$

- More than the number of atoms in observable universe

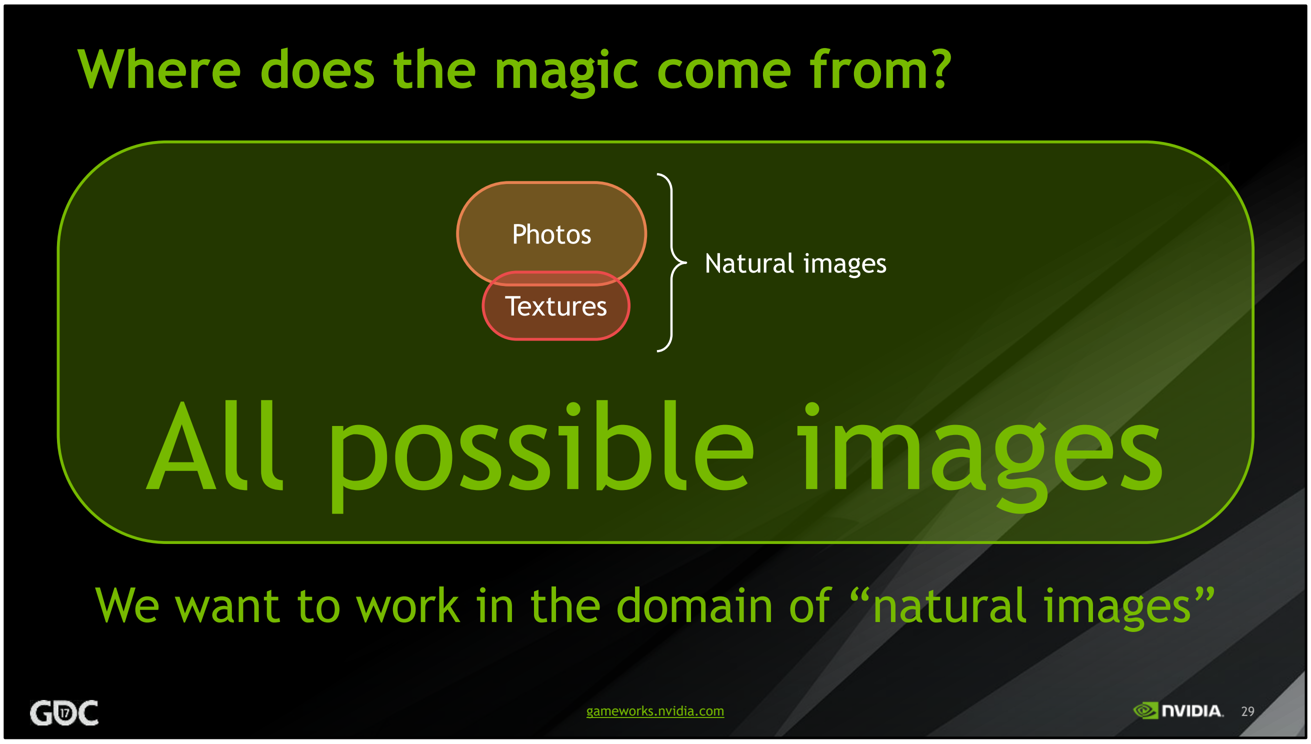
We don't need that much



gameworks.nvidia.com

28

That's actually more atoms than the observable universe, maybe an image contains less information than this.

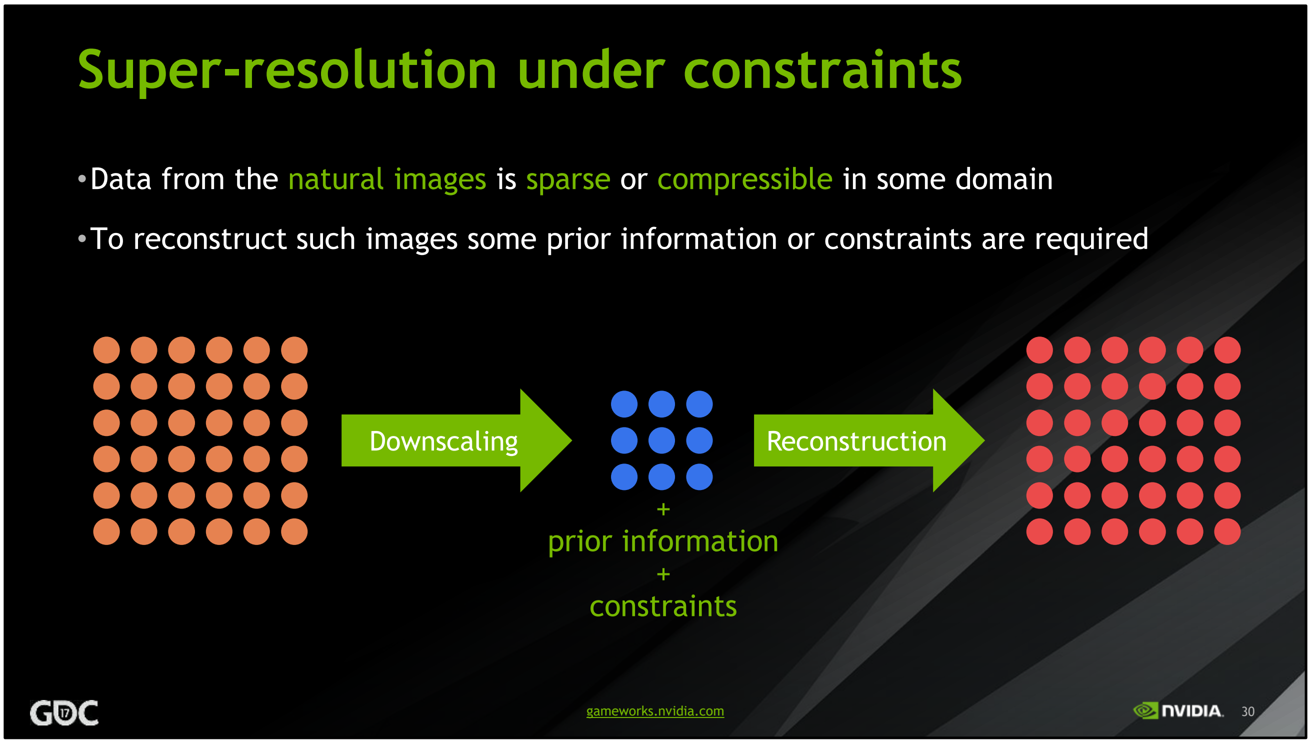


Where does the magic come from?

Photos
Textures
Natural images
All possible images

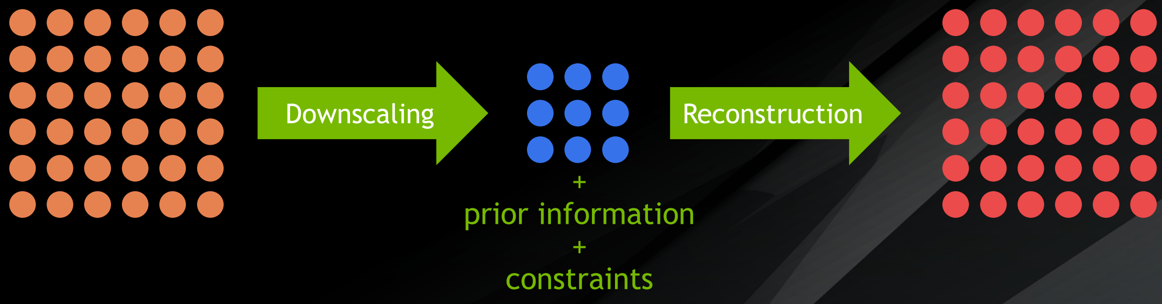
We want to work in the domain of "natural images"

Indeed, among the possible images, photos and textures are a very small subset



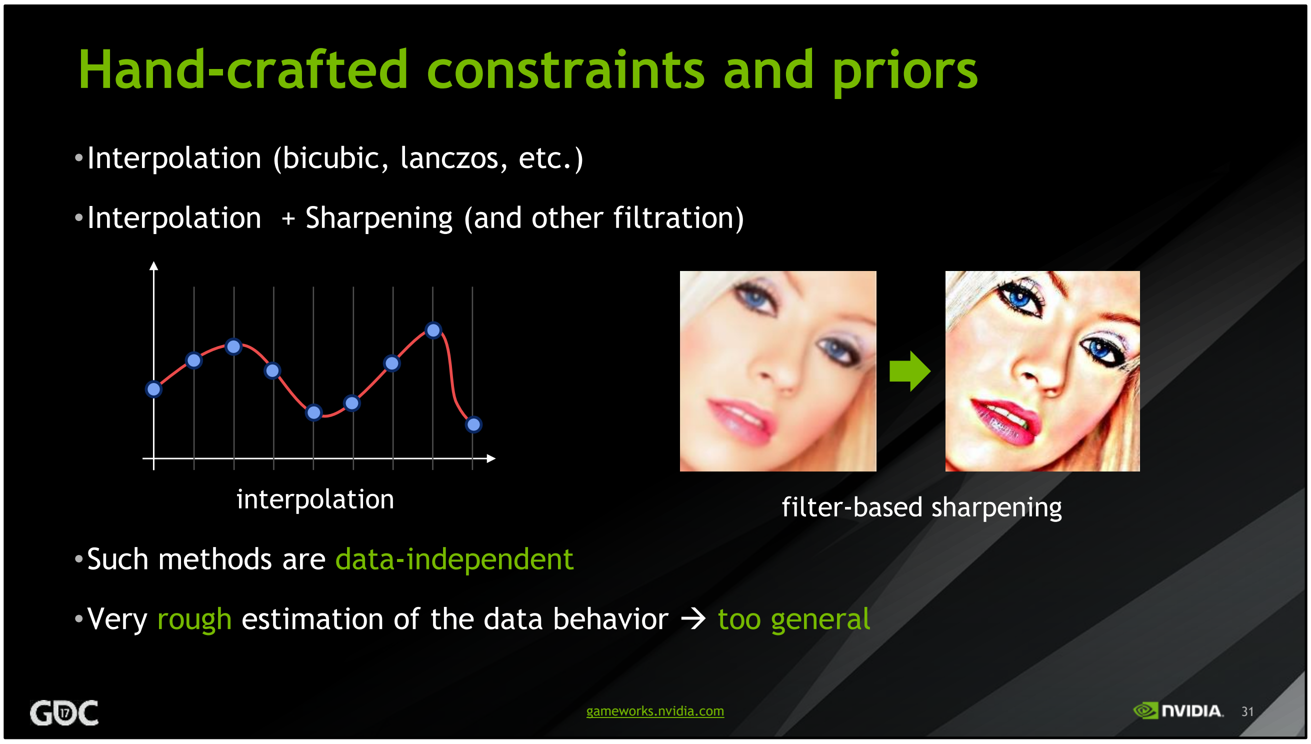
Super-resolution under constraints

- Data from the natural images is sparse or compressible in some domain
- To reconstruct such images some prior information or constraints are required



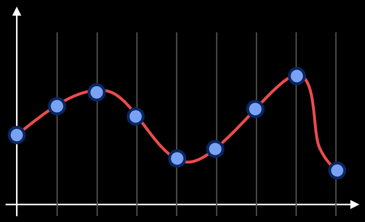
GDC gameworks.nvidia.com NVIDIA 30

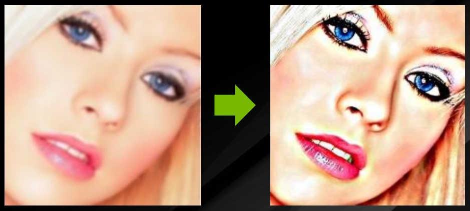
If we constrain our problem to deal with natural images and textures, we can be enhance the content without much loss



Hand-crafted constraints and priors

- Interpolation (bicubic, lanczos, etc.)
- Interpolation + Sharpening (and other filtration)


interpolation


filter-based sharpening

- Such methods are data-independent
- Very rough estimation of the data behavior → too general

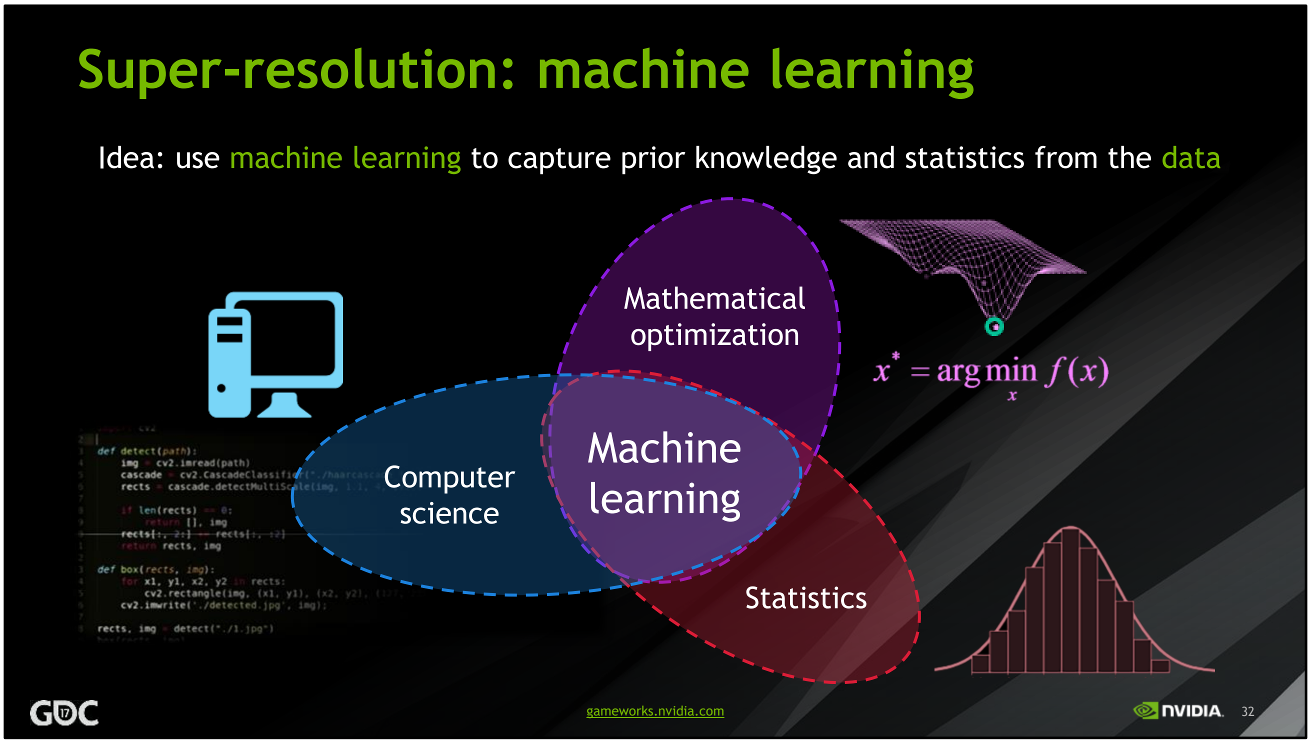
GDC gameworks.nvidia.com NVIDIA 31

One possible option is to construct an upscaling method taking some a priori decisions on the resulting image (e.g. sharpness)

This will work in some cases, but in general will require a lot manual of work to handmake the upscaling logics into our algorithm

We need a better method, something that looks into images from our specific domain and finds which are the interesting features.

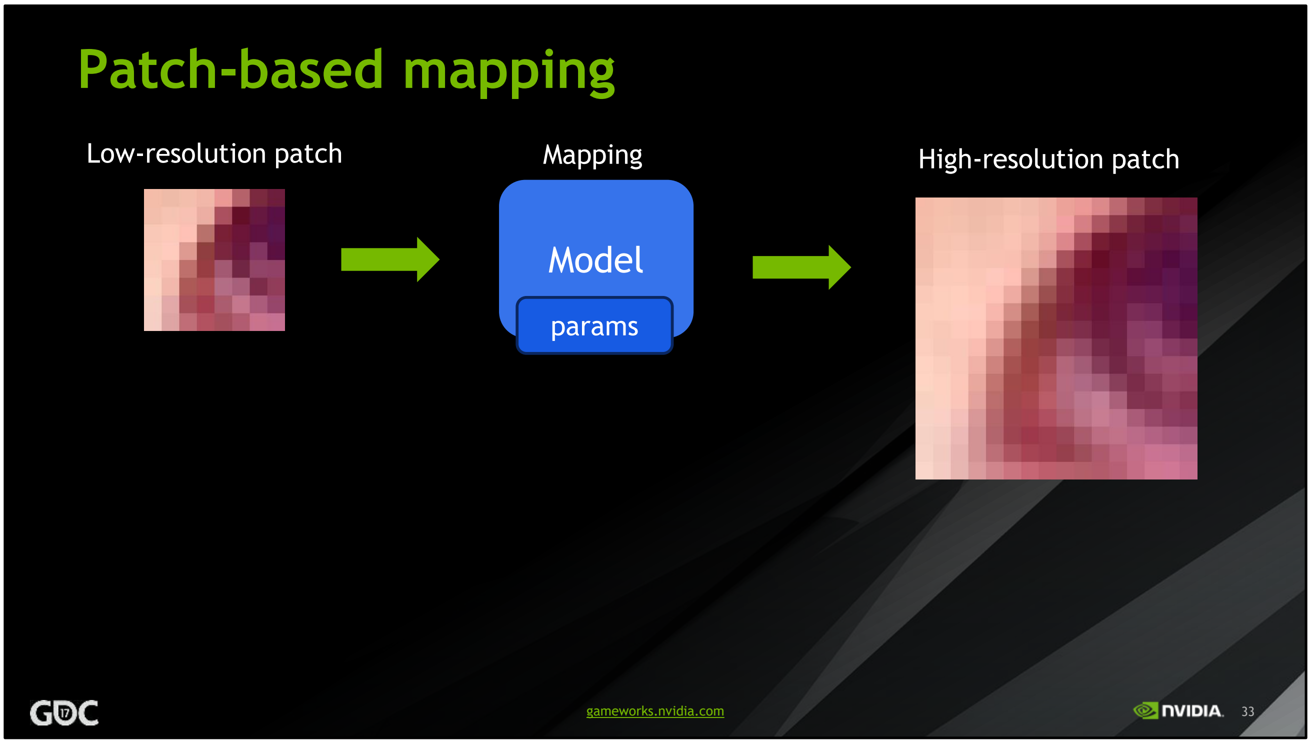
These methods are usually machine learning methods



Super-resolution: machine learning

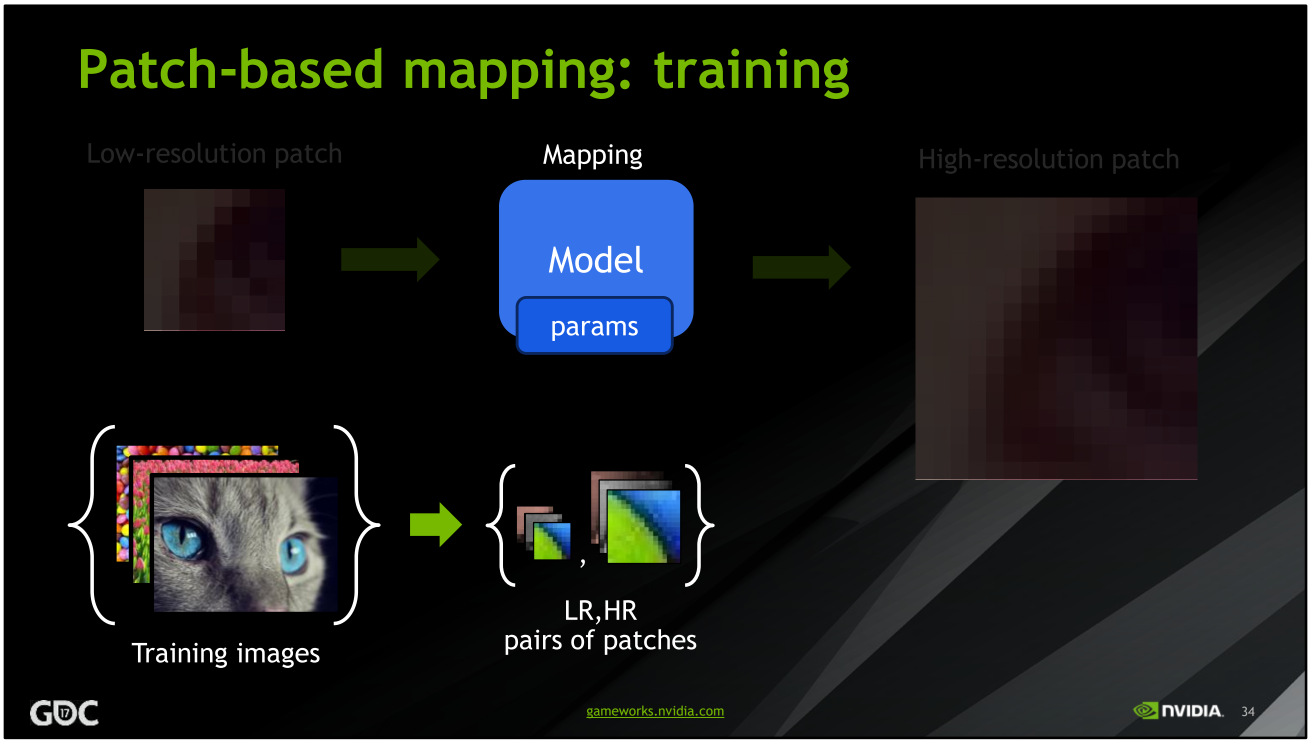
Idea: use machine learning to capture prior knowledge and statistics from the data

The idea is to exploit prior knowledge about our image domain. We can gather such information using machine learning. Since the machine learning is a technique of building intelligence systems, which are not explicitly programmed, but trained using an error minimization to capture and exploit internal structure and features of the training data automatically.



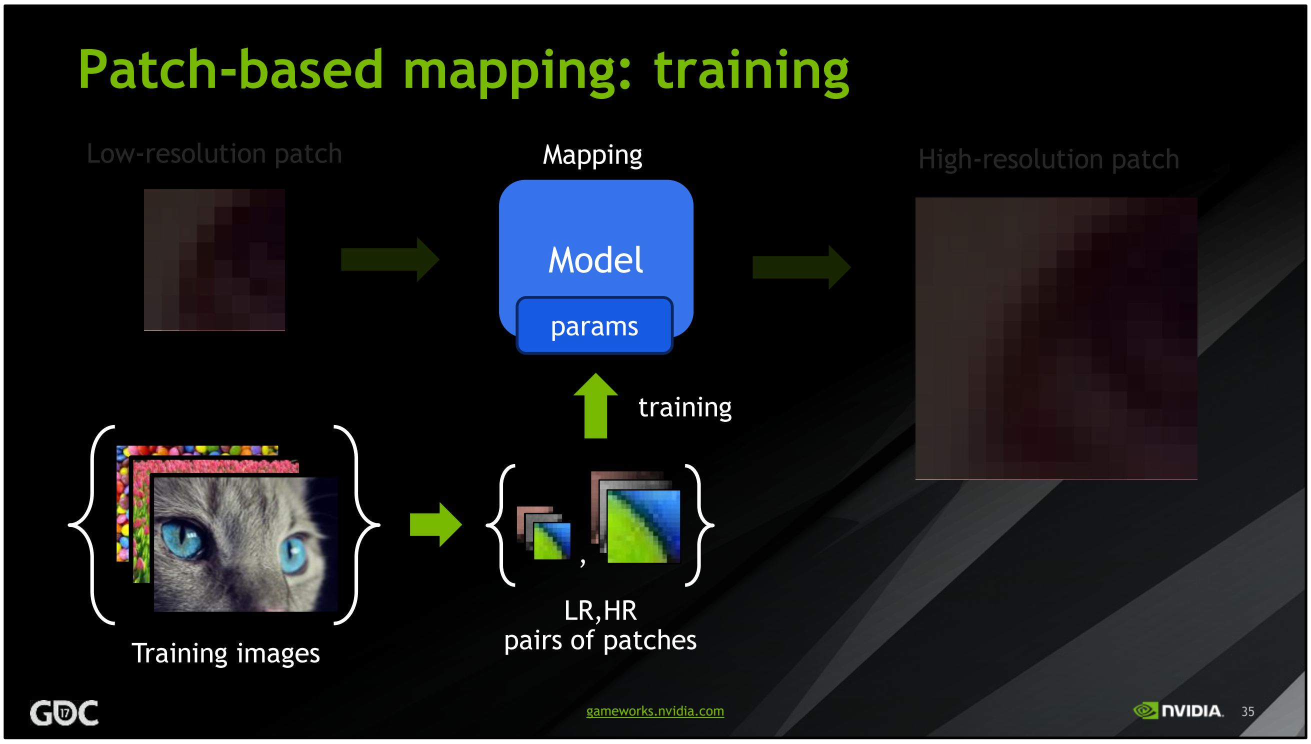
Patch-based mapping

Let's reduce our task to a simpler one: transformation of an image patch. Let's consider a mapping function, which constructs high-resolution patch by a given low-resolution patch from the input image. Such mapping function will depend on a set of parameters, which we want to find using machine learning.



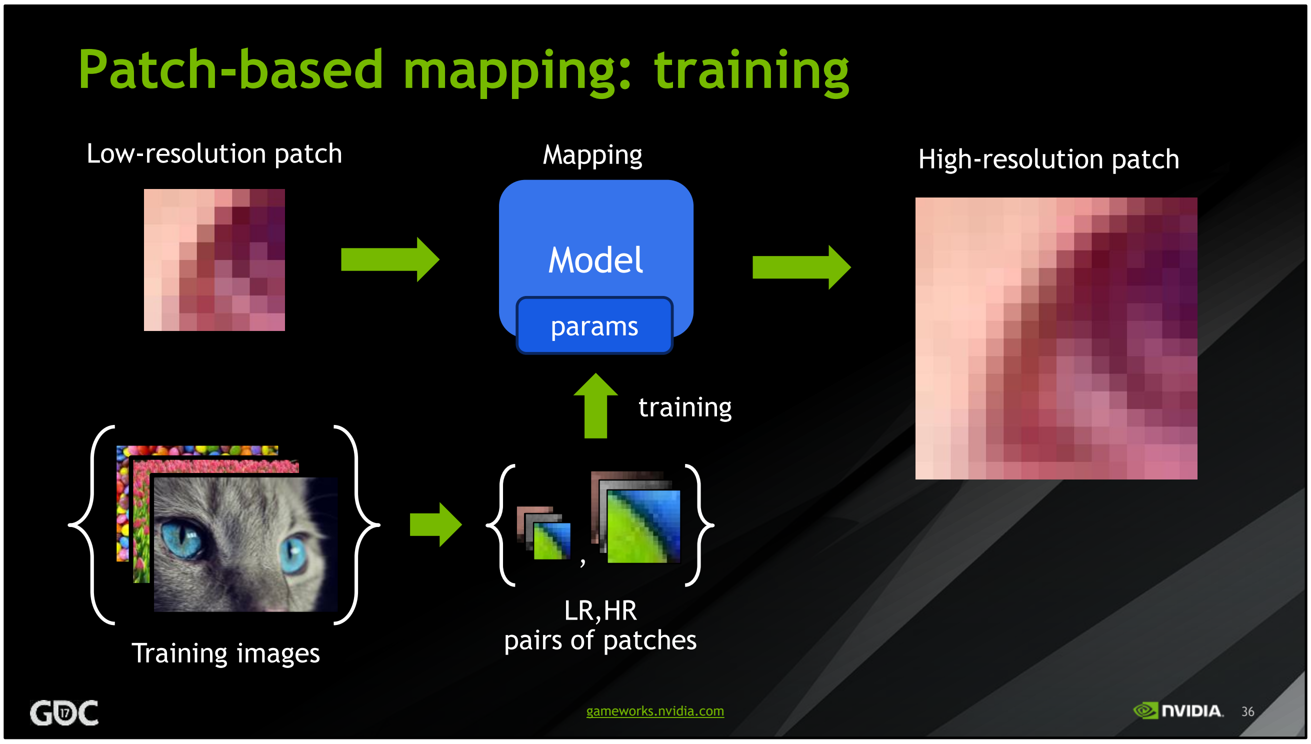
Patch-based mapping: training

We are training our model in a supervised fashion. So we need to collect a set of pairs of low-resolution and corresponding ground-truth high-resolution patches, what could be easily done if we have a set of high-resolution images.



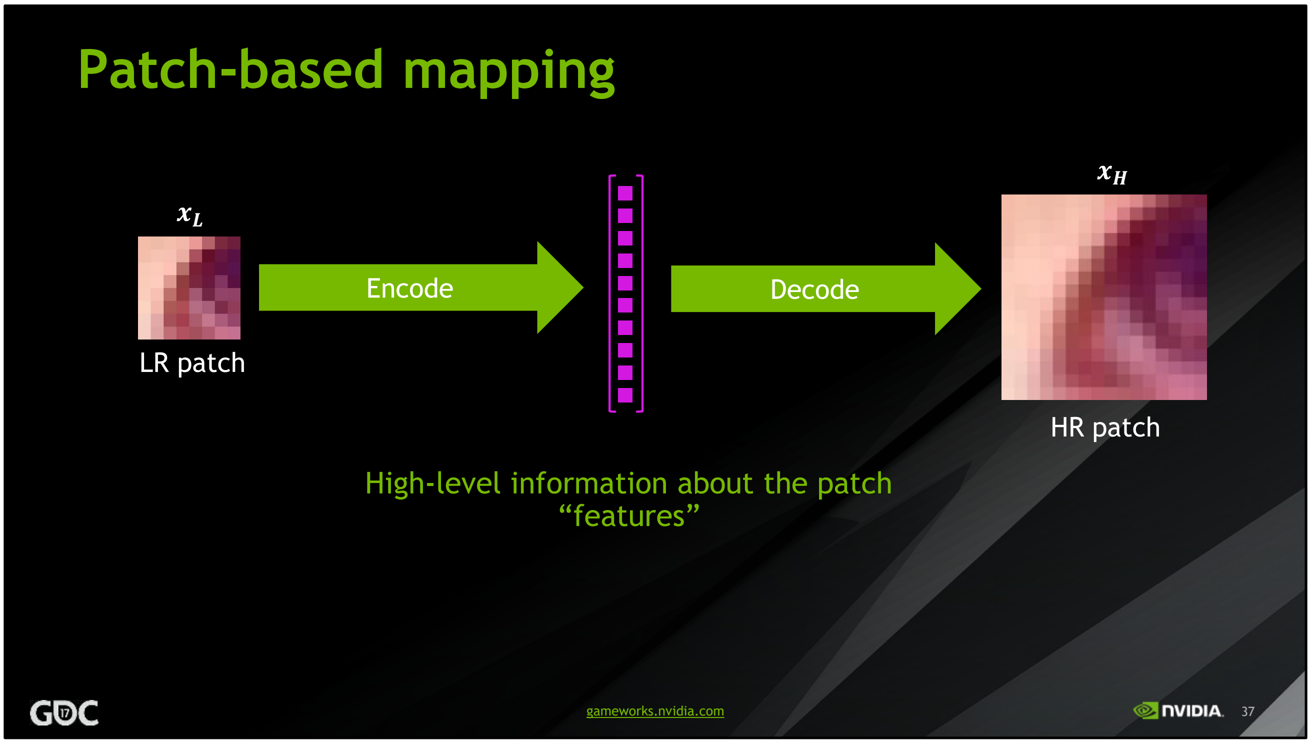
Patch-based mapping: training

And we pass that set of pairs into the training process,



Patch-based mapping: training

after which we expect that our model will be capable to predict high-resolution patch in the most optimal way.

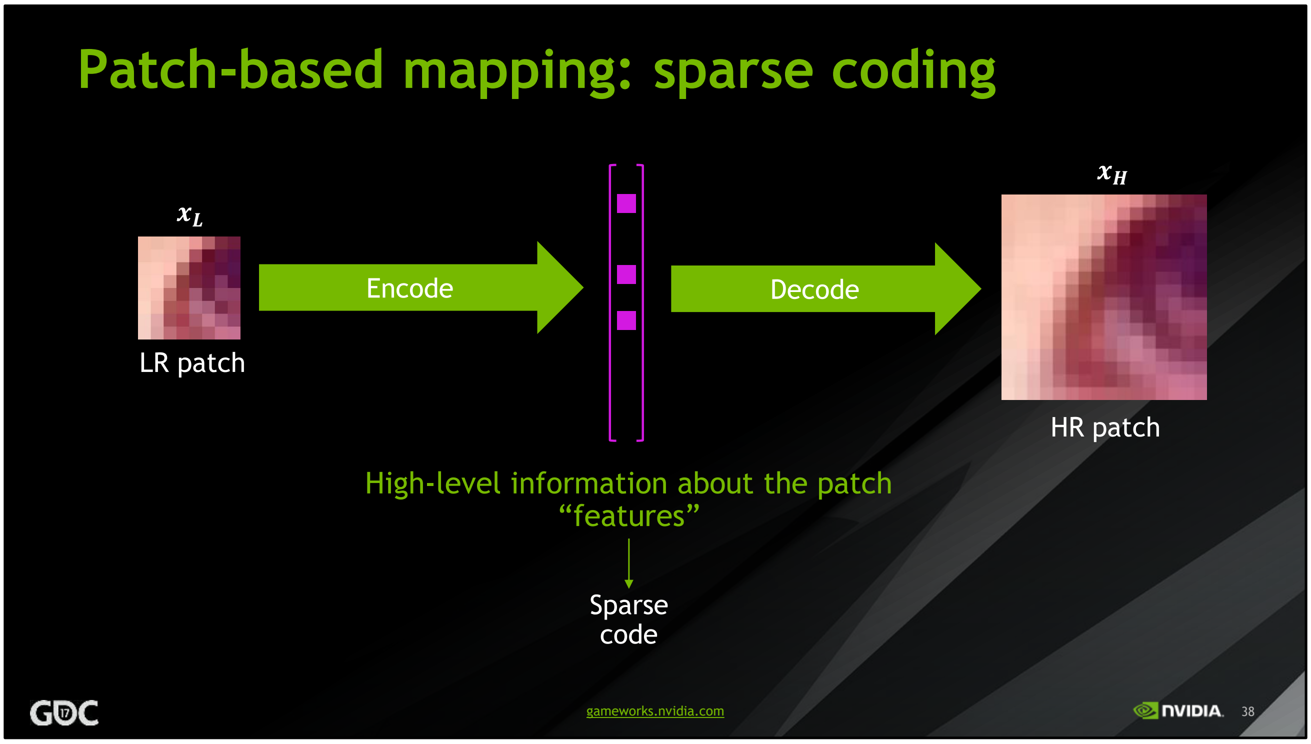


Patch-based mapping

gameworks.nvidia.com

37

A good way to build the mapping function is to use an encoding of an input patch into some intermediate scale-invariant representation, which will carry some semantic information about the patch.

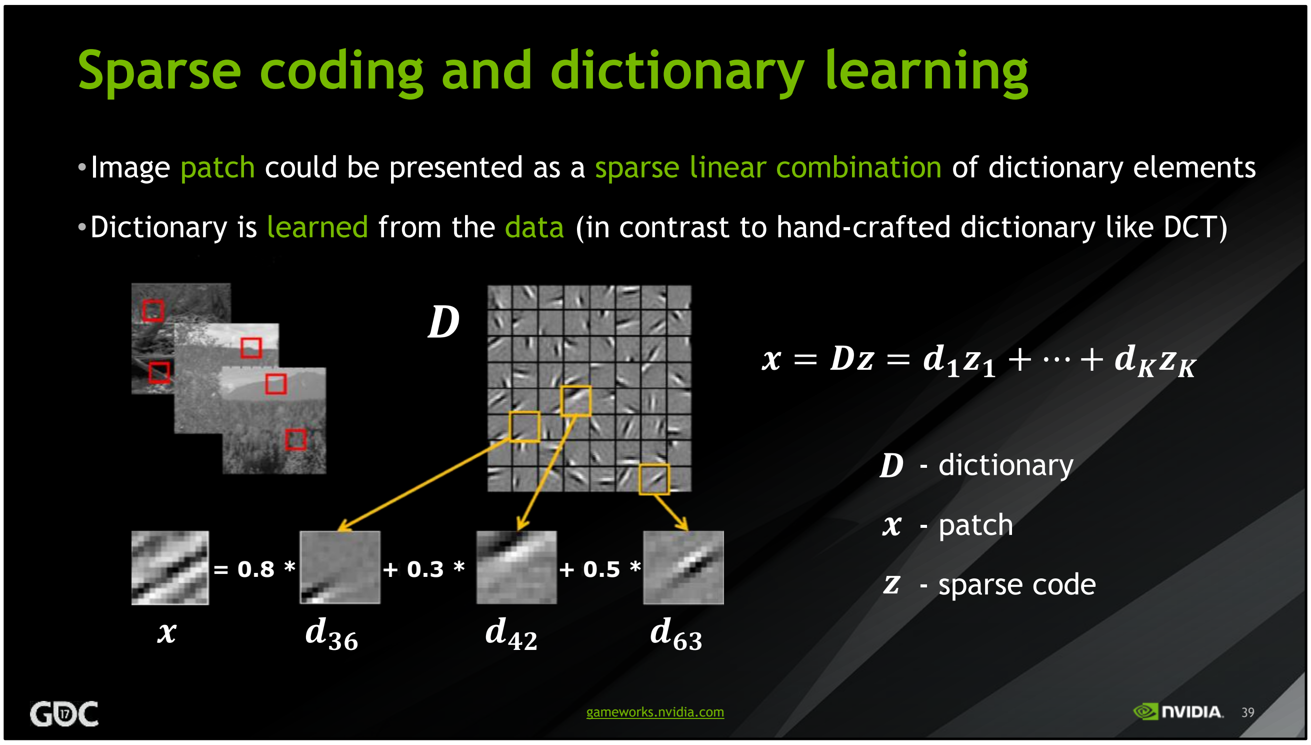


Patch-based mapping: sparse coding

gameworks.nvidia.com

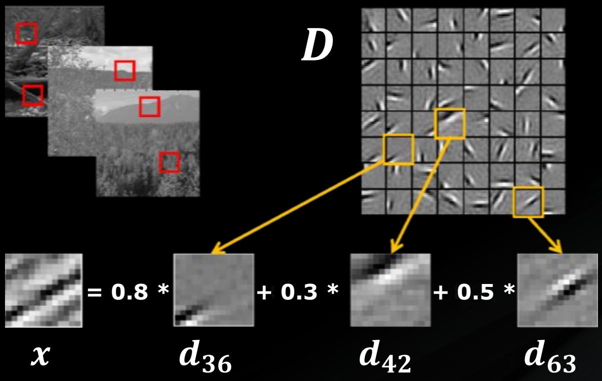
38

One way to build such representation is sparse coding. Here we exploit our prior knowledge, that our signal is sparse in some domain.



Sparse coding and dictionary learning

- Image patch could be presented as a sparse linear combination of dictionary elements
- Dictionary is learned from the data (in contrast to hand-crafted dictionary like DCT)



$$x = Dz = d_1 z_1 + \cdots + d_K z_K$$

D - dictionary

x - patch

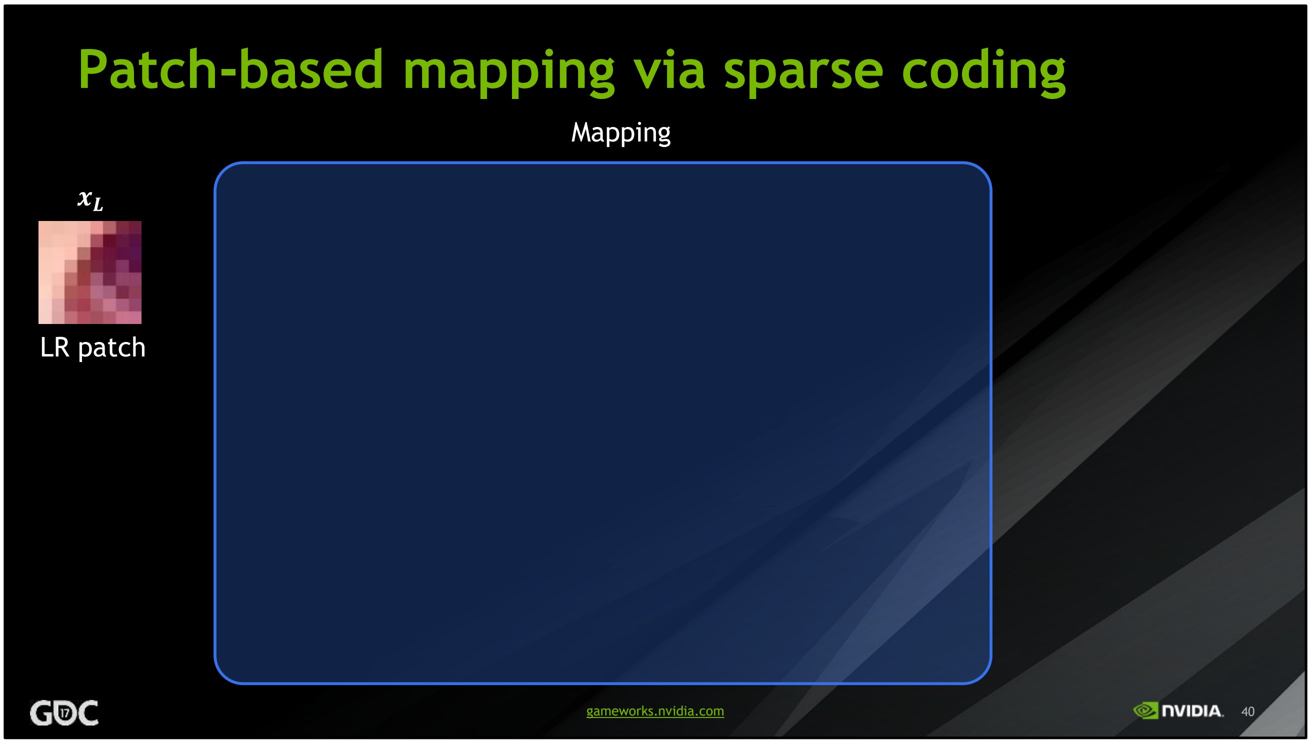
z - sparse code

GDC 17

gameworks.nvidia.com

NVIDIA 39

In particular, we assume that the patch could be represented as a linear combination of only a small number of elements from some dictionary. Using that dictionary, we can obtain corresponding coefficients (also known as sparse codes, carrying high-level representation) and vise-versa.



Patch-based mapping via sparse coding

Mapping

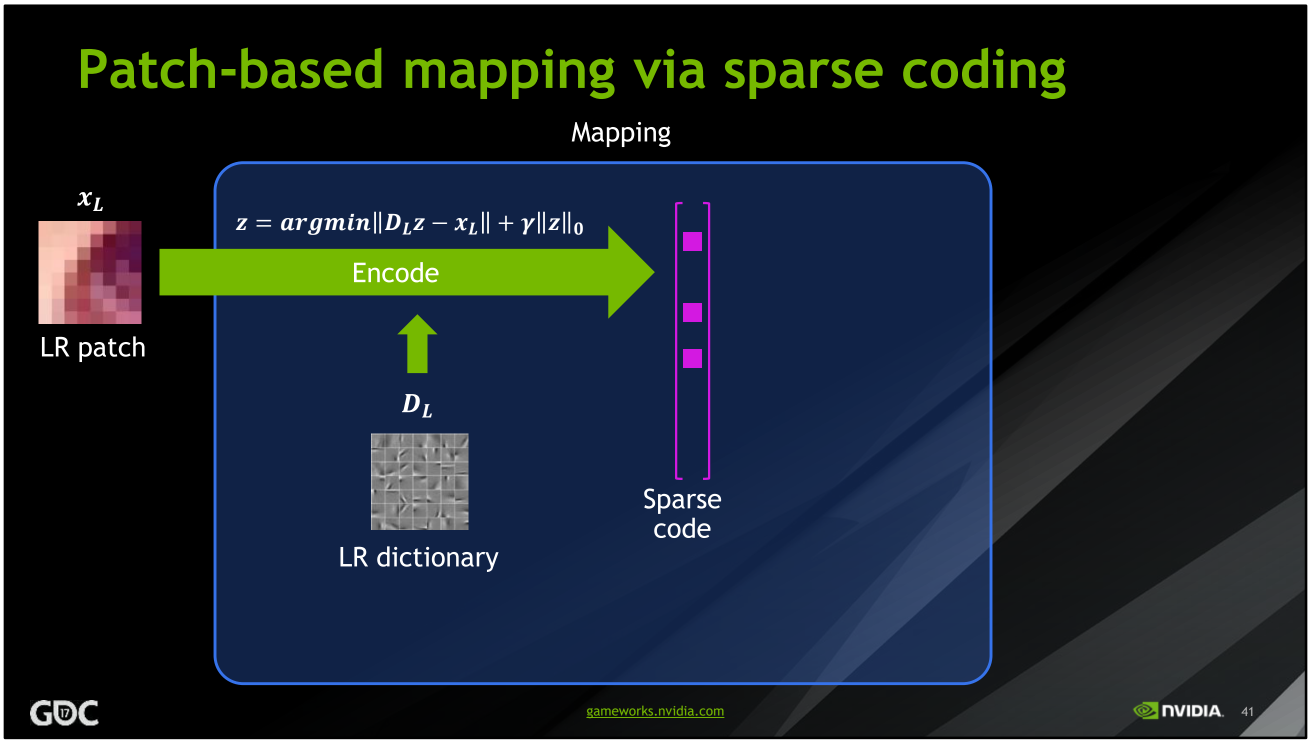
x_L

LR patch

gameworks.nvidia.com

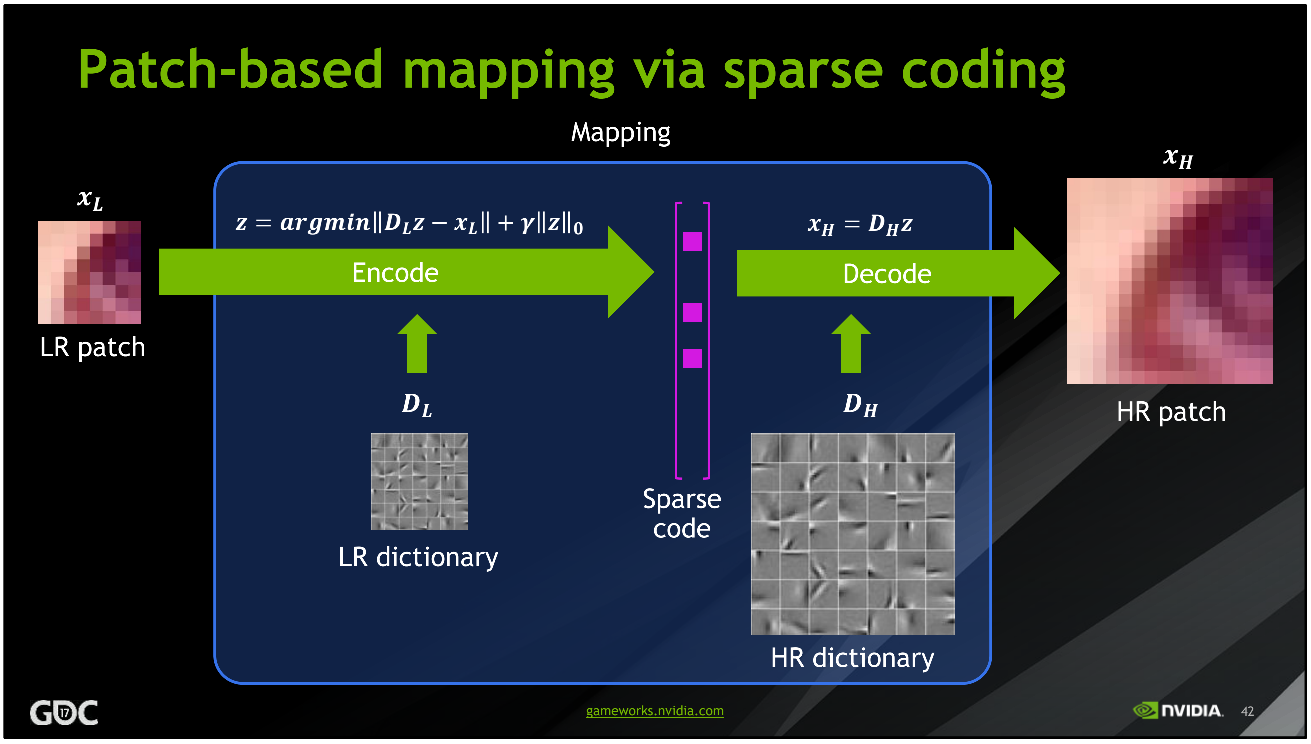
40

How could it be used for super resolution?
Given a low-resolution patch....



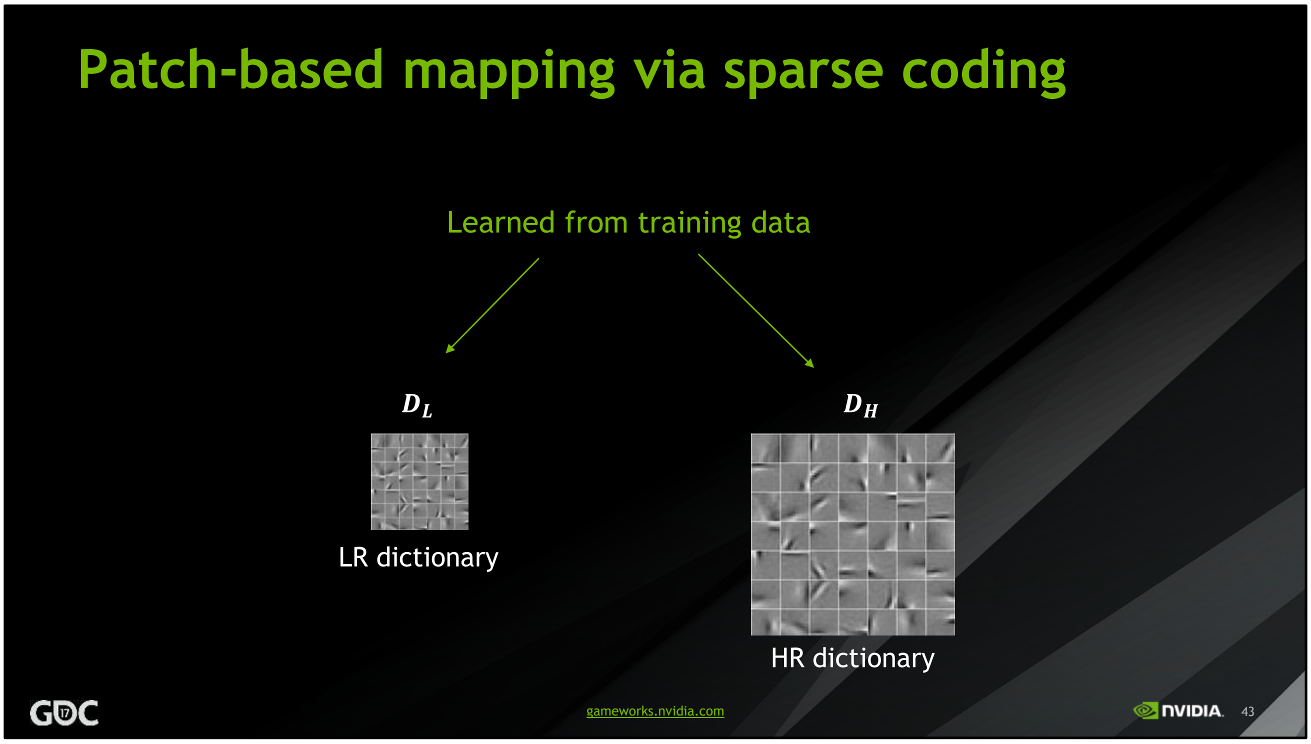
Patch-based mapping via sparse coding

and low-resolution dictionary, we perform the sparse encoding (using some optimization procedure).



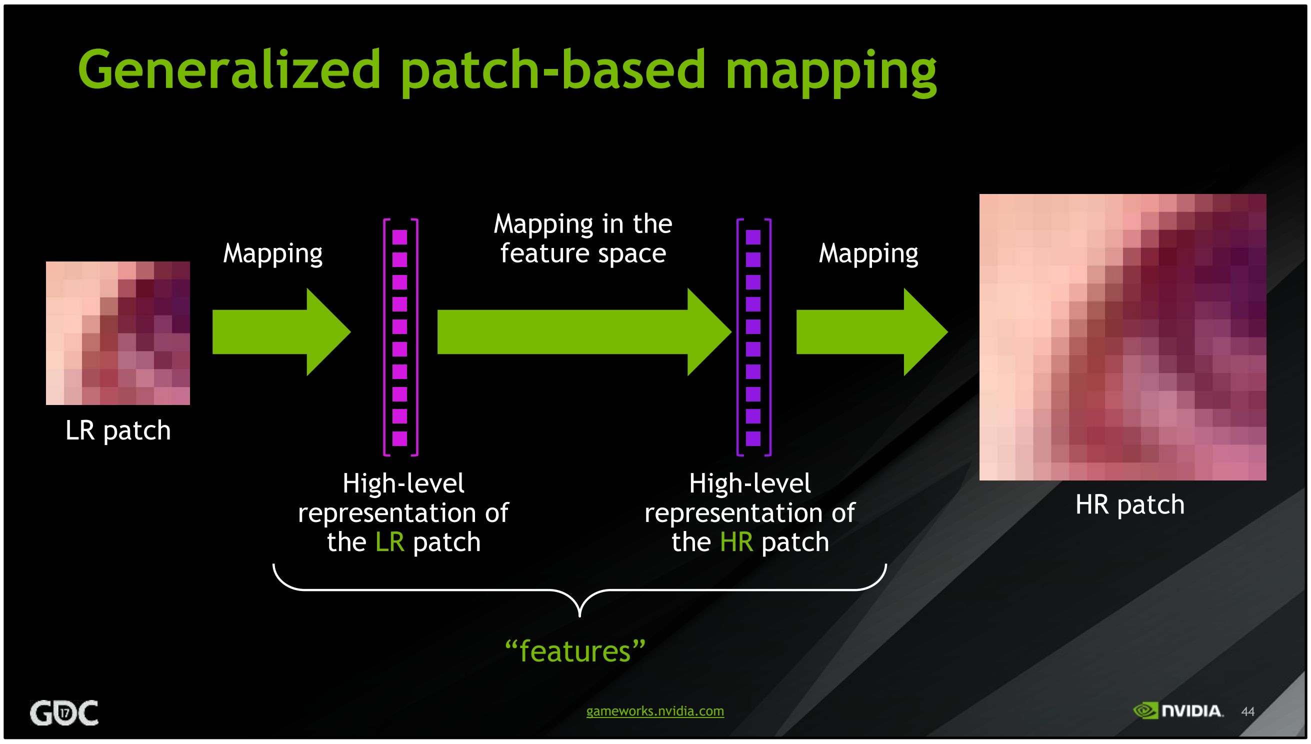
Patch-based mapping via sparse coding

Then, given the sparse codes and high-resolution dictionary, we perform decoding, simply calculating the linear combination.



Patch-based mapping via sparse coding

We train the dictionaries to capture internal structure of the signal by maximizing the sparsity of the encoding.



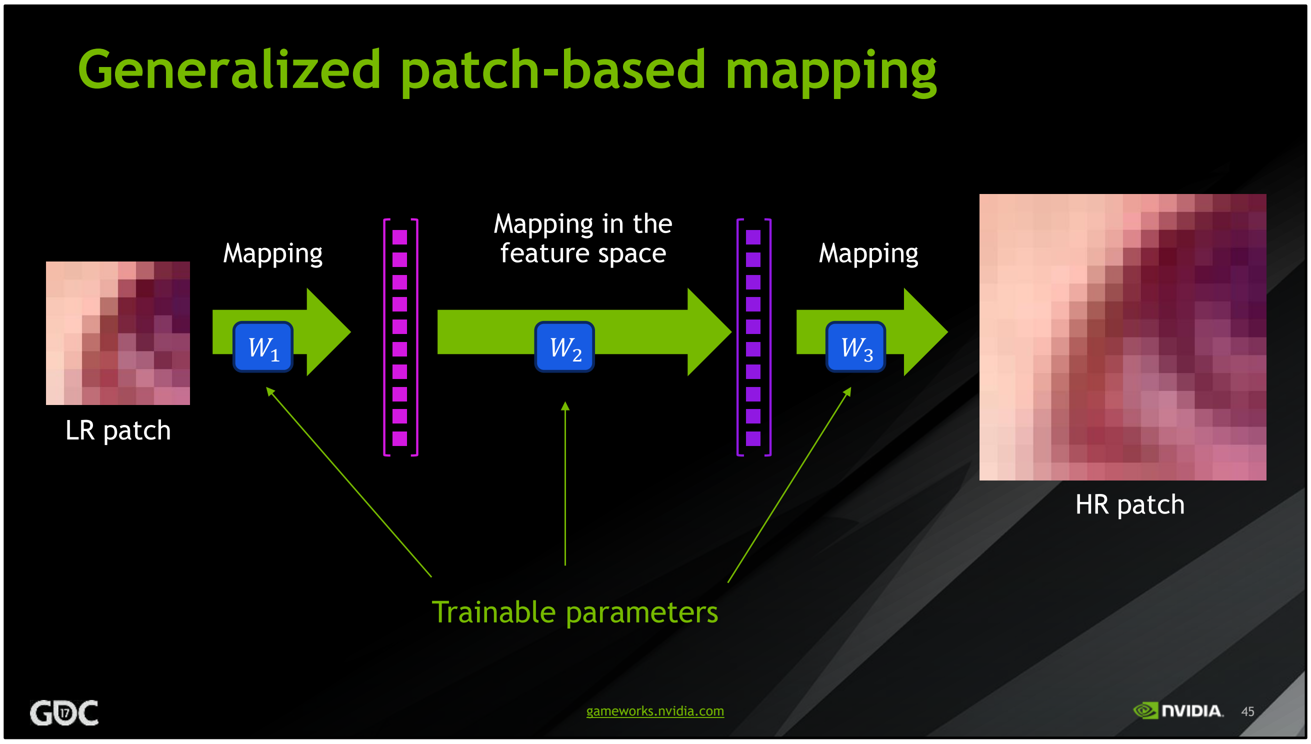
Generalized patch-based mapping

LR patch
Mapping
Mapping in the feature space
Mapping
HR patch
High-level representation of the LR patch
High-level representation of the HR patch
"features"

gameworks.nvidia.com

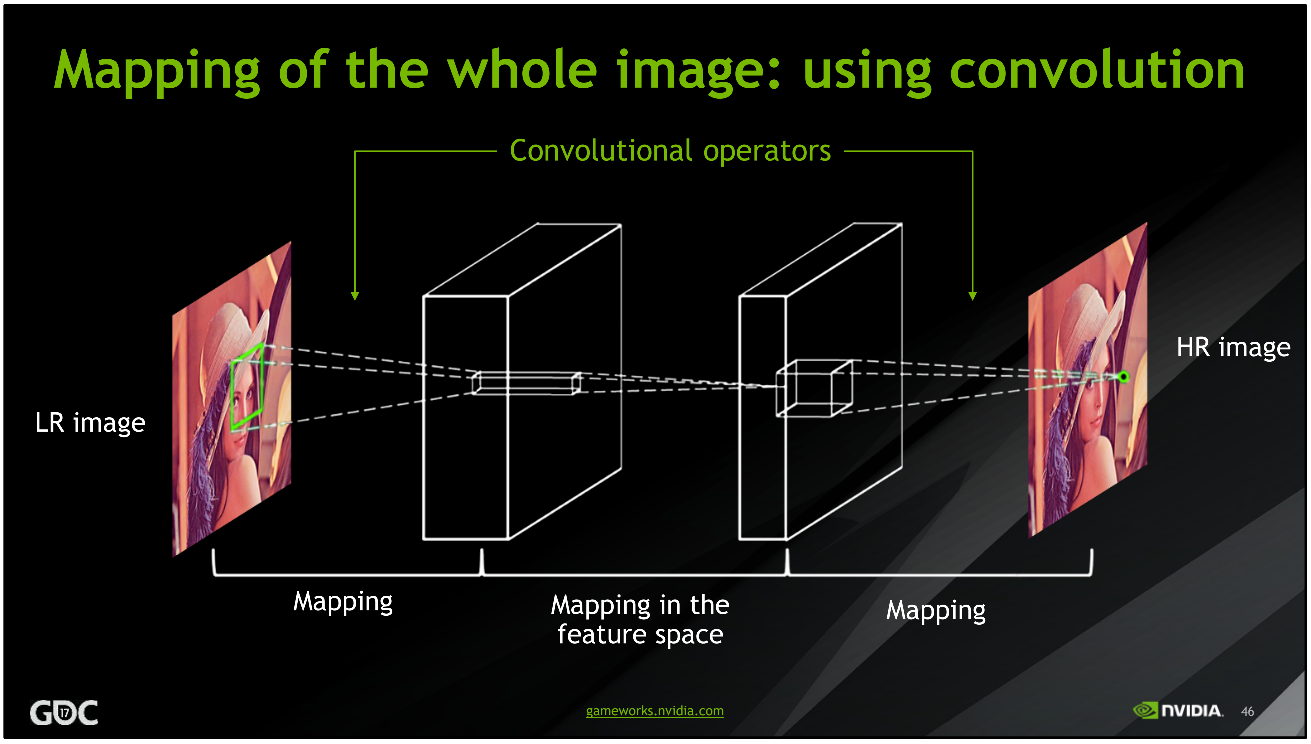
44

We may generalize the idea and build another mapping function with more complex internal representation. For example, first map input patch into corresponding high-level representation. Then perform some transformation in that space. And then map resulting high-level representation back to image space -- to high-resolution patch.



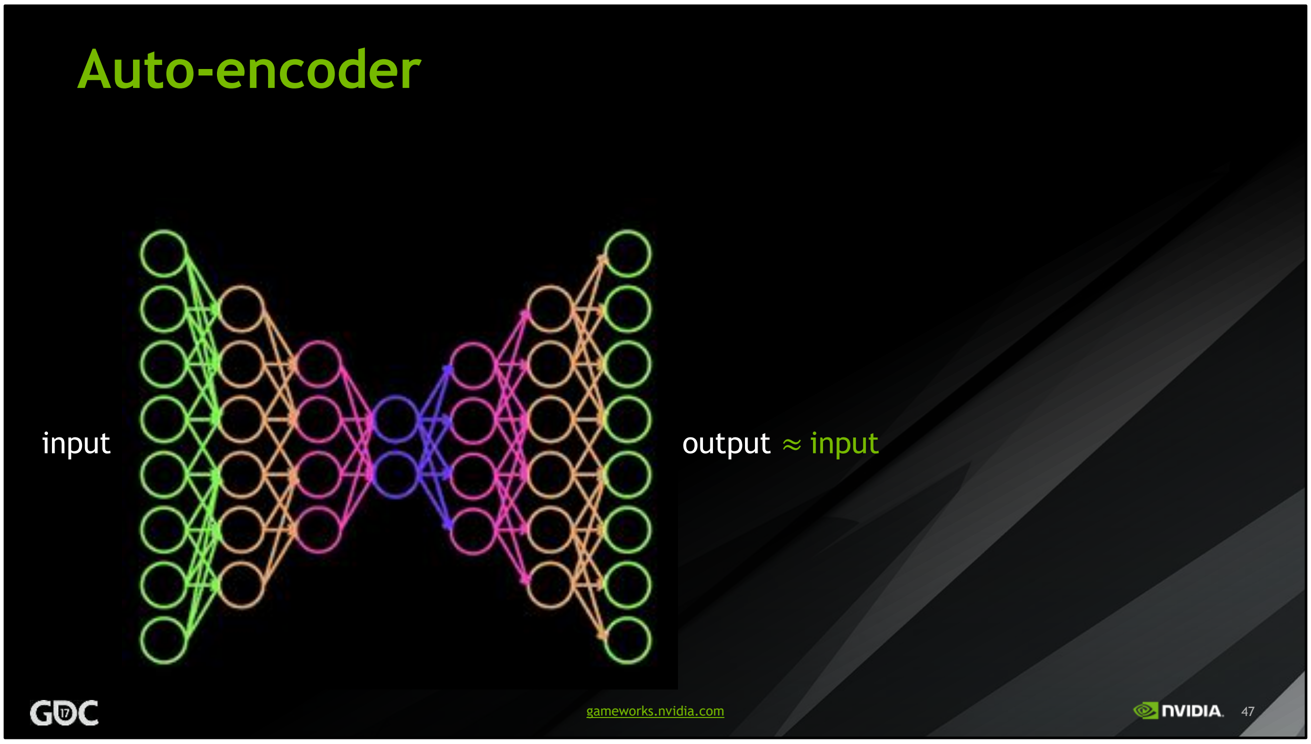
Generalized patch-based mapping

All transformations depend on some parameters, which we adjust during the training. This could be a neural net, for example.

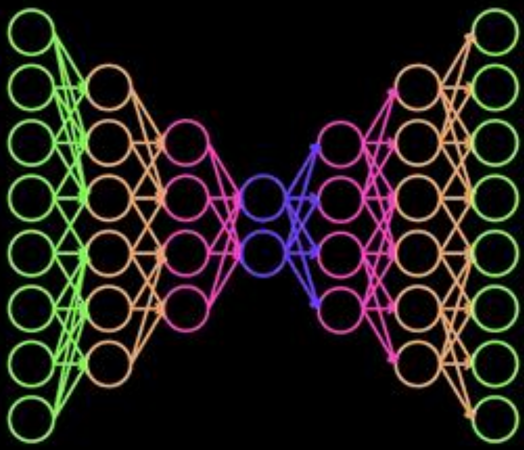


Mapping of the whole image: using convolution

Now let's recall that we actually want to do a super-resolution for the whole image. In this case, we can apply our patch-based transformation to the set of all overlapping patches on the input image, and then assemble resulting high-resolution patches into high-resolution output. These operations could be implemented via a convolutional operator. And presented structure is very similar to one well-known type of neural networks -- auto-encoders.



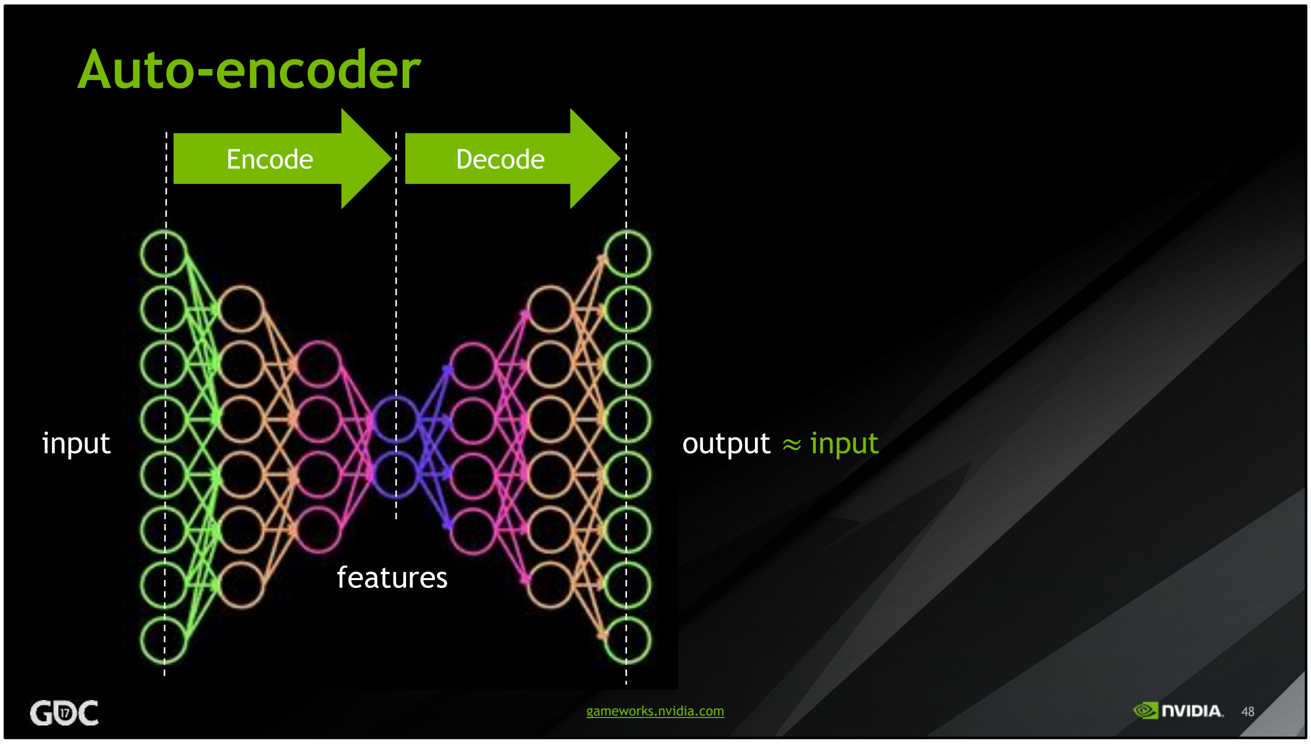
Auto-encoder



What's an Auto-Encoder?

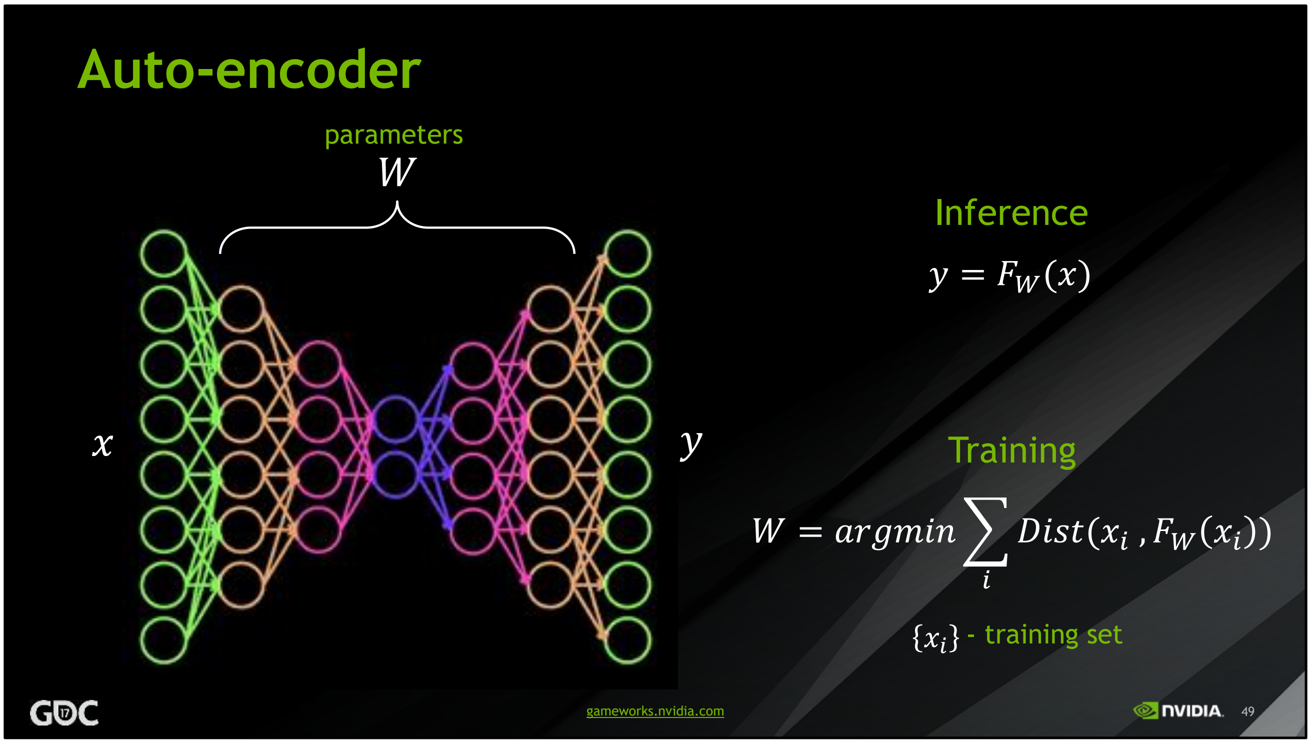
It's a neural network trained to reconstruct its input.

What's difficult is doing it by passing to an internal representation, with less information (hourglass structure)

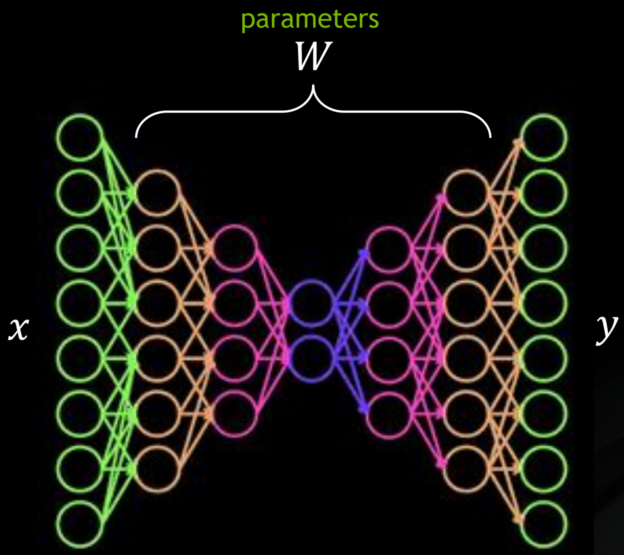


Auto-encoder

An autoencoder network is composed by two parts, an ENCODER which take the input and converts it to the internal representation (feature space) and a DECODER which tries to regenerate the input



Auto-encoder



Inference

$$y = F_W(x)$$

Training

$$W = argmin \sum_i Dist(x_i\,, F_W(x_i))$$

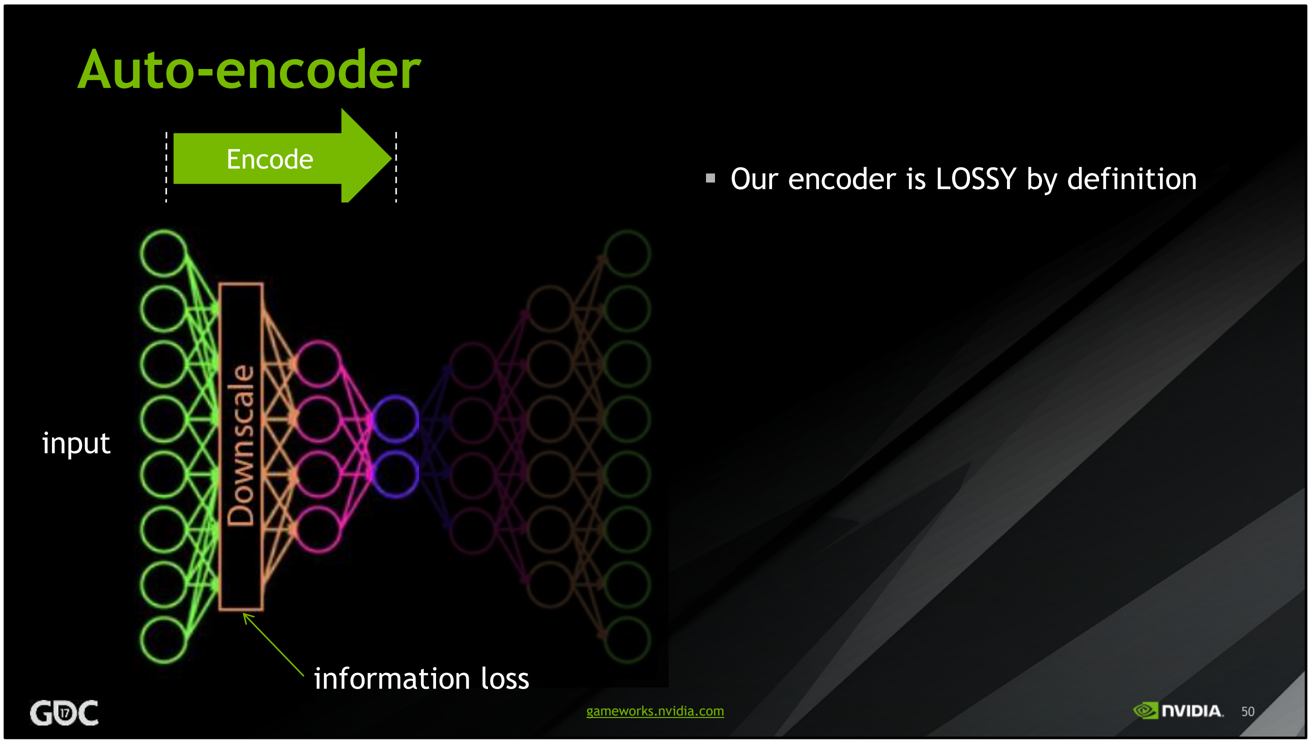
$\{x_i\}$ - training set

gameworks.nvidia.com

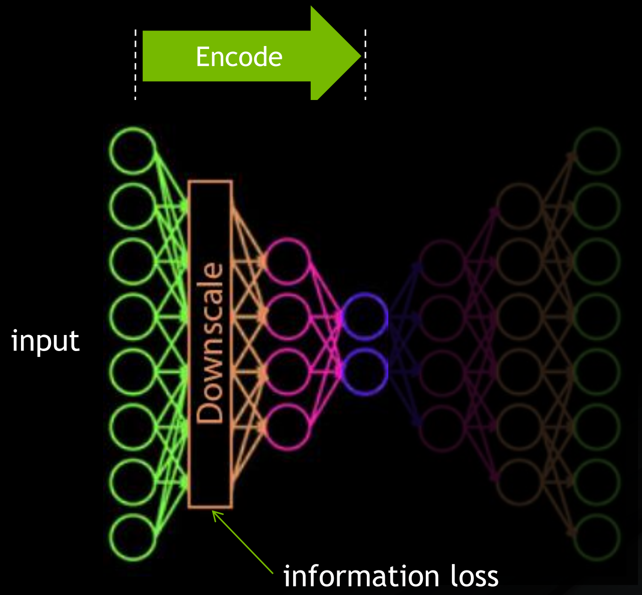
49

When encoder and decoder are modeled by a DNN, the parameter space is defined by a set of Weights (W).

In the training we try to minimize a specific loss function (or “distance” between the input and the output). If there’s enough information in the middle layer + in the prior knowledge, the reconstruction will be perfect (distance will be 0), if there isn’t enough information, the network will try to minimize the distance measured on the training set.



Auto-encoder

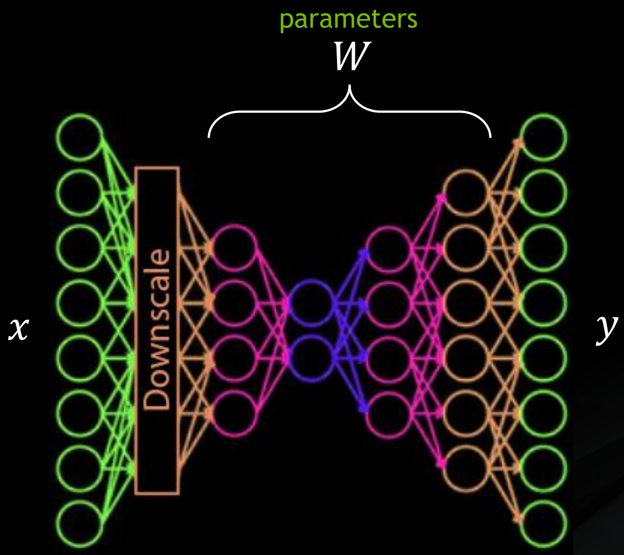


- Our encoder is LOSSY by definition

gameworks.nvidia.com 50

In our case, we KNOW the internal representation is LOSSY because we explicitly introduced a downscale layer (which removes information) when creating our encoder.

Super-resolution auto-encoder



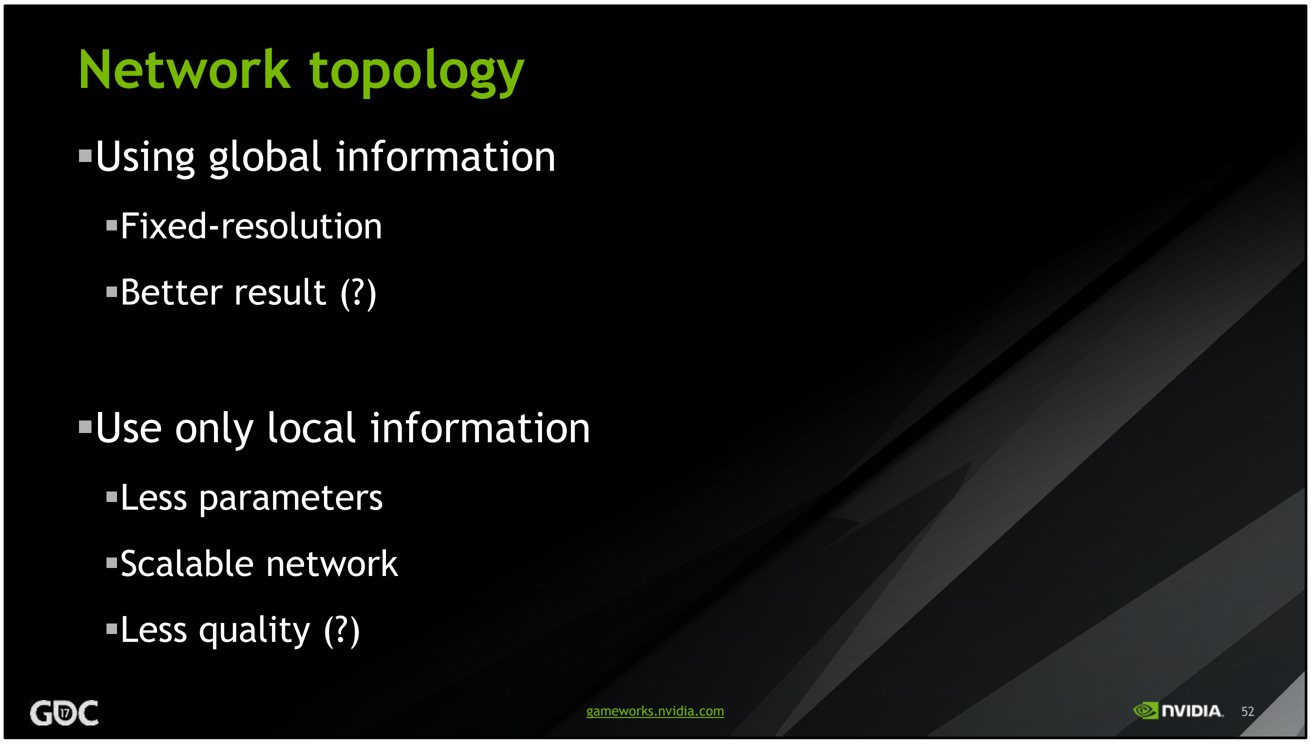
Inference

$$y = F_W(x)$$

Training

$$W = argmin \sum_i Dist(x_i , F_W(x_i))$$

$\{x_i\}$ - training set



Network topology

- Using global information
  - Fixed-resolution
  - Better result (?)
- Use only local information
  - Less parameters
  - Scalable network
  - Less quality (?)

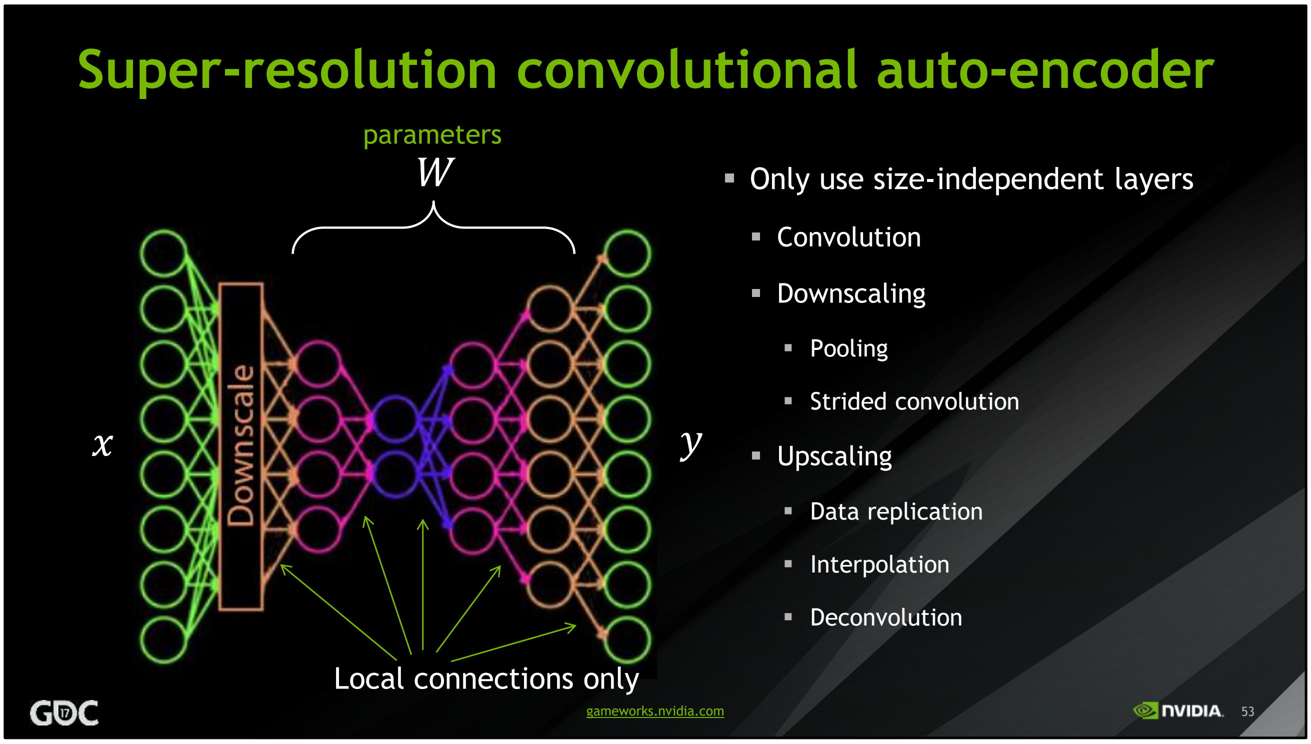
GDC 17

gameworks.nvidia.com

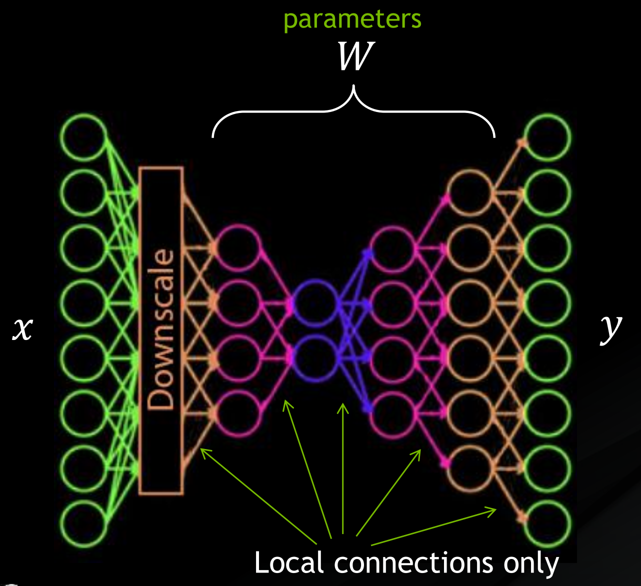
NVIDIA 52

Using all pixels in the image. Does this means having better results? Maybe.

Using only local information we have less parameters, a scalable network. Does this mean less quality? Not necessarily, we are using LOCAL information.

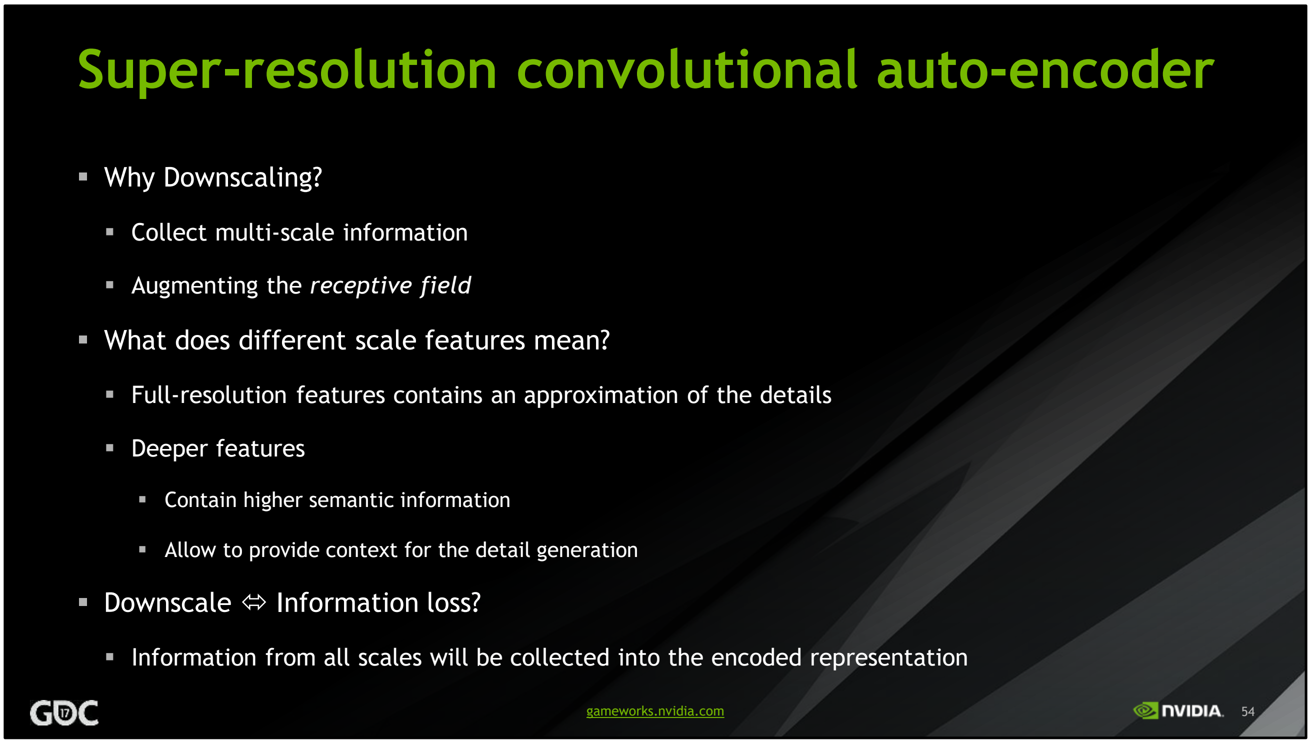


Super-resolution convolutional auto-encoder



- Only use size-independent layers
  - Convolution
  - Downscaling
    - Pooling
    - Strided convolution
  - Upscaling
    - Data replication
    - Interpolation
    - Deconvolution

gameworks.nvidia.com

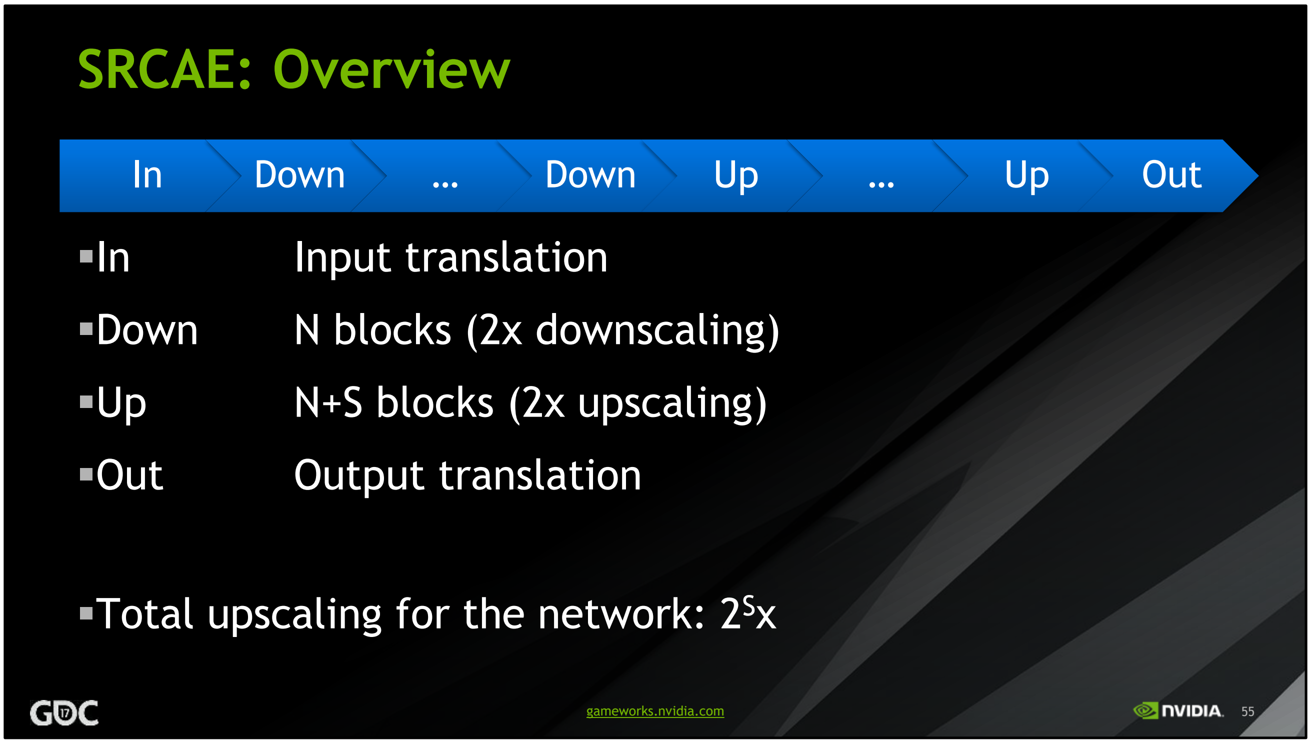

Super-resolution convolutional auto-encoder

- Why Downscaling?
  - Collect multi-scale information
  - Augmenting the *receptive field*
- What does different scale features mean?
  - Full-resolution features contains an approximation of the details
  - Deeper features
    - Contain higher semantic information
    - Allow to provide context for the detail generation
- Downscale ⇔ Information loss?
  - Information from all scales will be collected into the encoded representation

GDC gameworks.nvidia.com NVIDIA 54

Definition of receptive field.

Deeper downscaled layers contains more feature (channels)



SRCAE: Overview

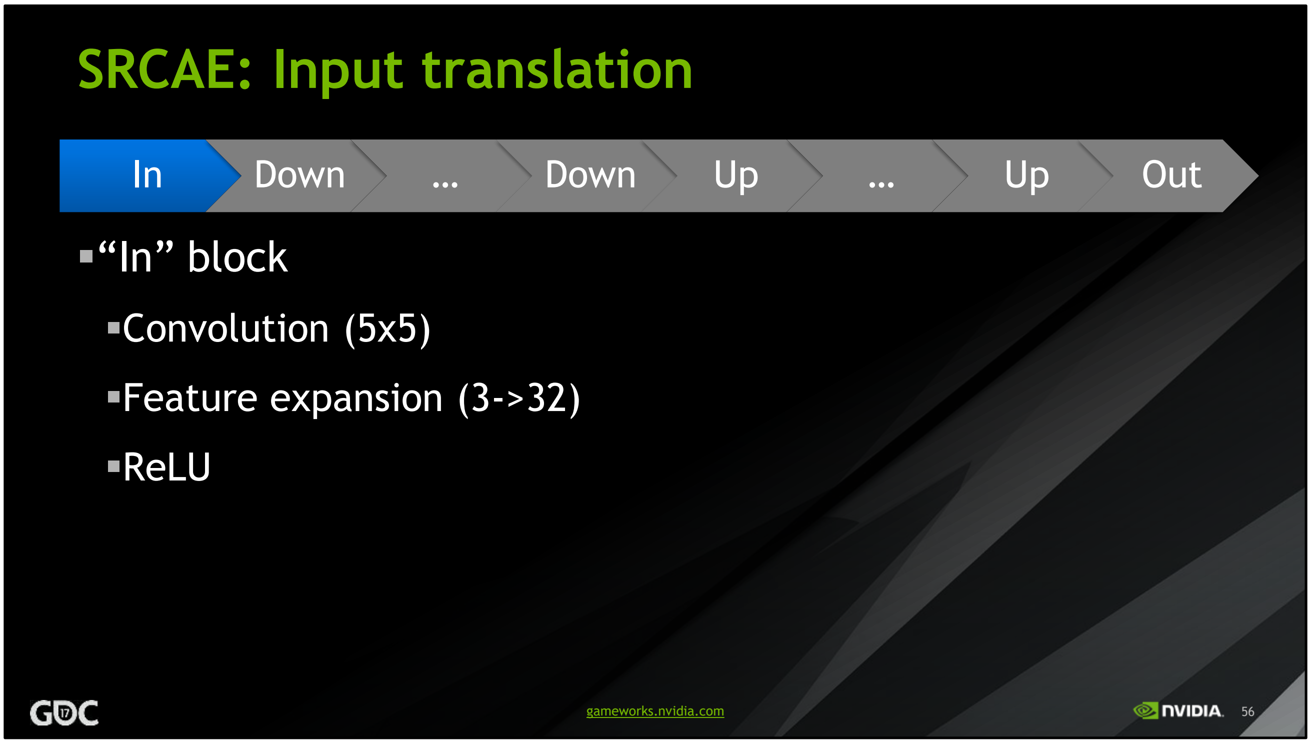
In → Down → … → Down → Up → … → Up → Out

▪In	Input translation
▪Down	N blocks (2x downscaling)
▪Up	N+S blocks (2x upscaling)
▪Out	Output translation

▪Total upscaling for the network: 2^Sx

gameworks.nvidia.com

55



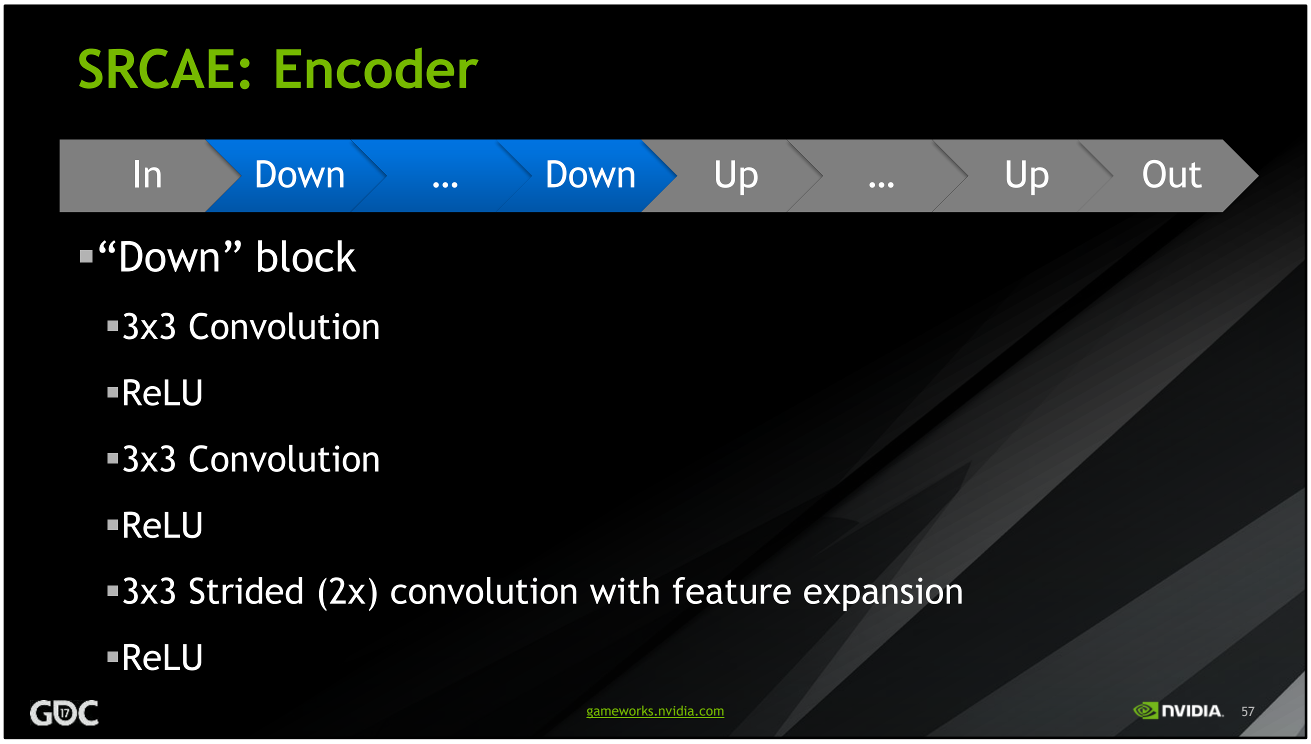
SRCAE: Input translation

In → Down → … → Down → Up → … → Up → Out

- "In" block
  - Convolution (5x5)
  - Feature expansion (3->32)
  - ReLU

gameworks.nvidia.com

56

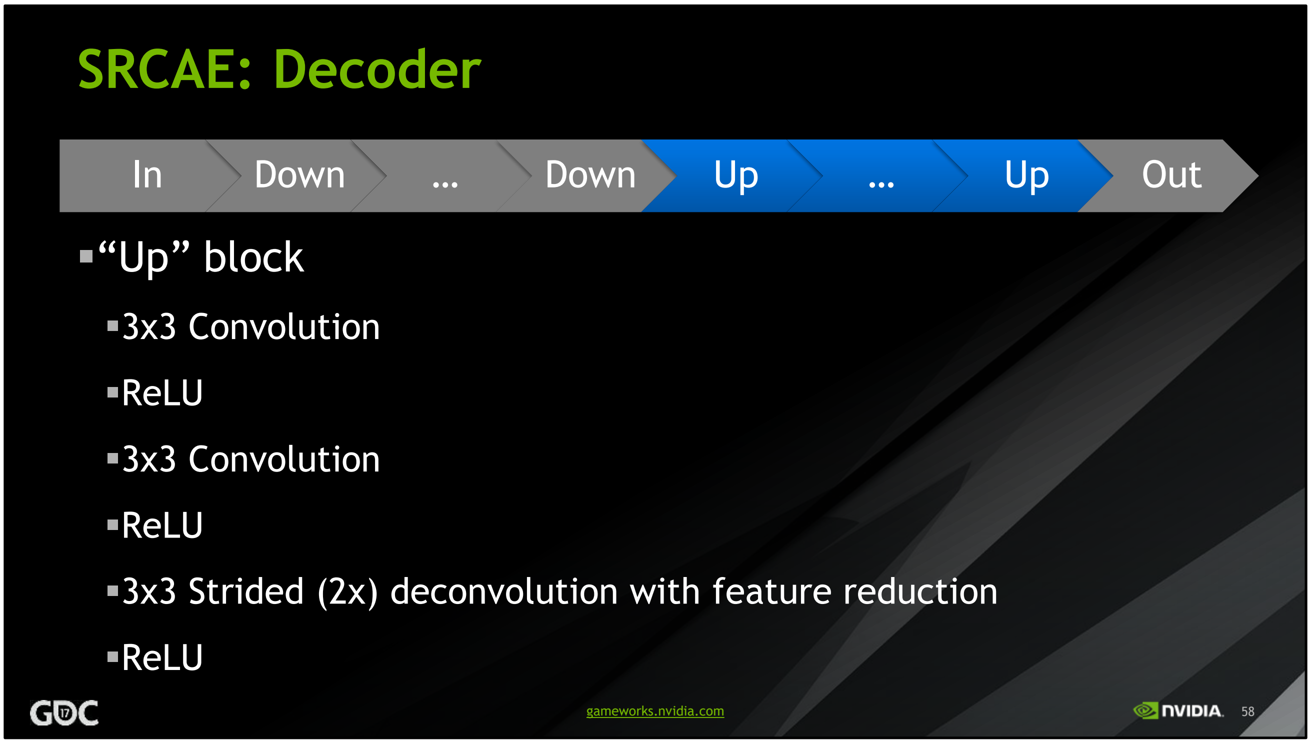


SRCAE: Encoder

In → Down → … → Down → Up → … → Up → Out

- "Down" block
  - 3x3 Convolution
  - ReLU
  - 3x3 Convolution
  - ReLU
  - 3x3 Strided (2x) convolution with feature expansion
  - ReLU

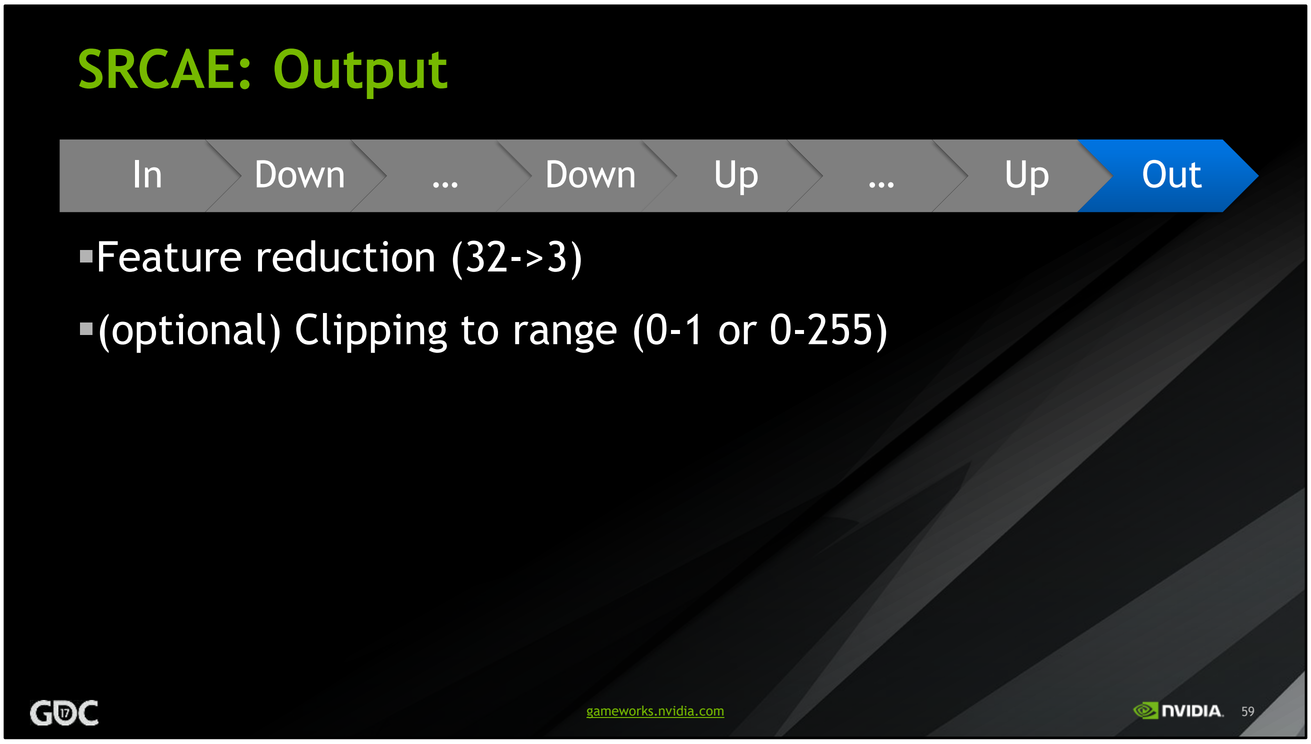
GDC gameworks.nvidia.com NVIDIA 57



SRCAE: Decoder

In	Down	…	Down	Up	…	Up	Out

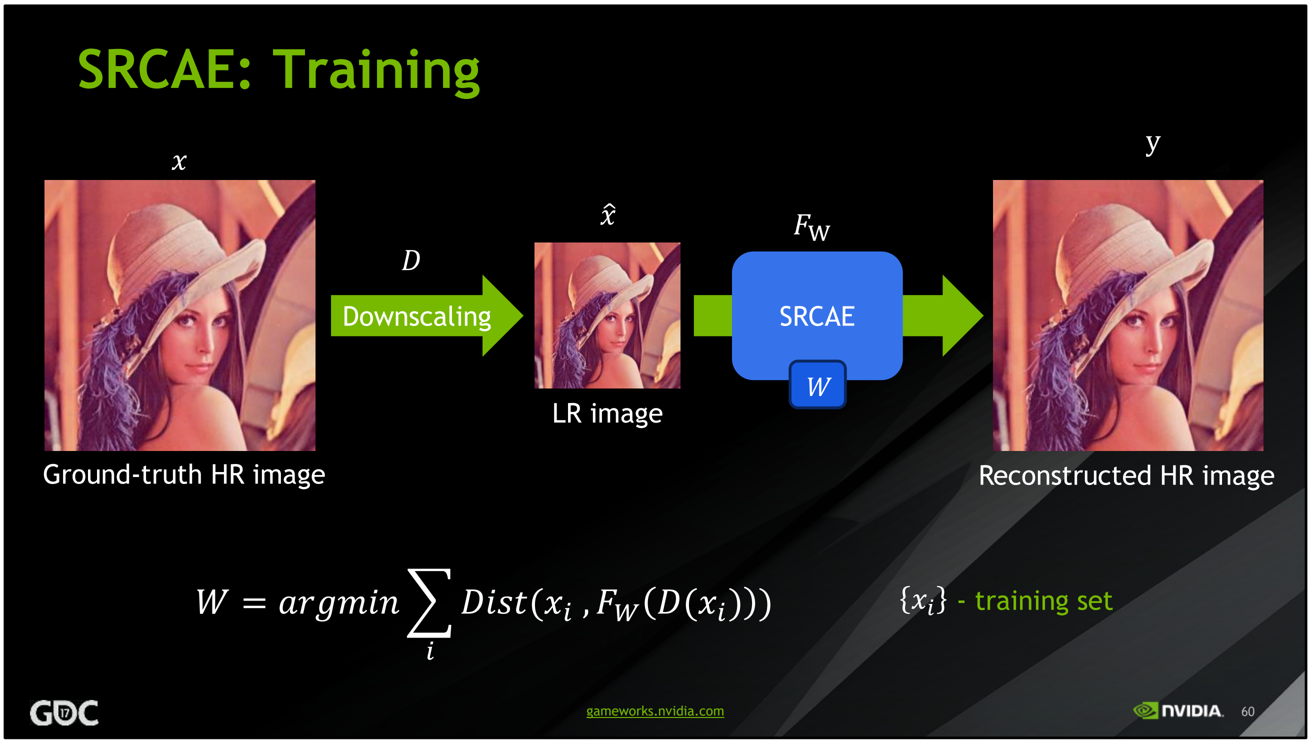
- "Up" block
  - 3x3 Convolution
  - ReLU
  - 3x3 Convolution
  - ReLU
  - 3x3 Strided (2x) deconvolution with feature reduction
  - ReLU



SRCAE: Output

In
Down
...
Down
Up
...
Up
Out

- Feature reduction (32->3)
- (optional) Clipping to range (0-1 or 0-255)

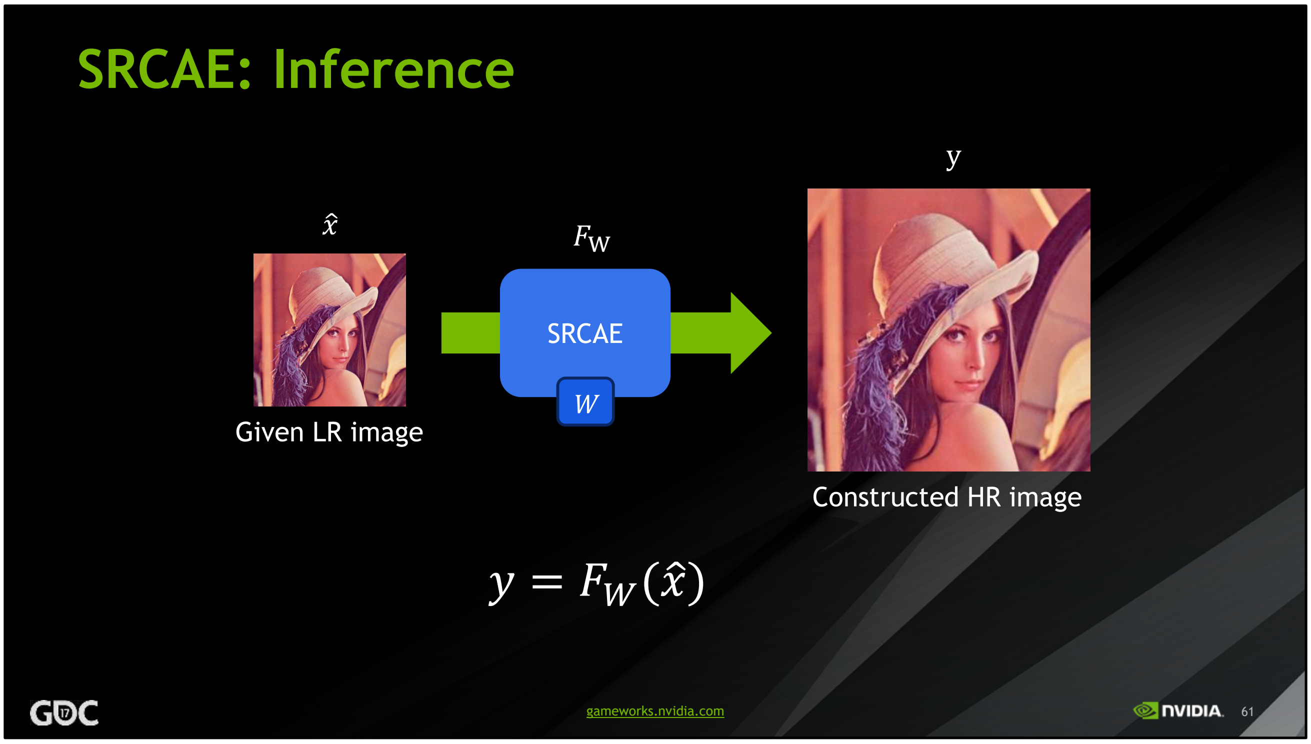


SRCAE: Training

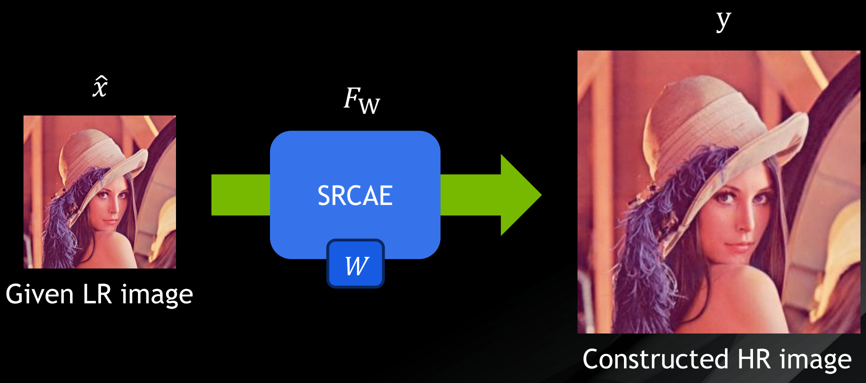
$$W = argmin \sum_i Dist(x_i, F_W(D(x_i)))$$

$\{x_i\}$ - training set

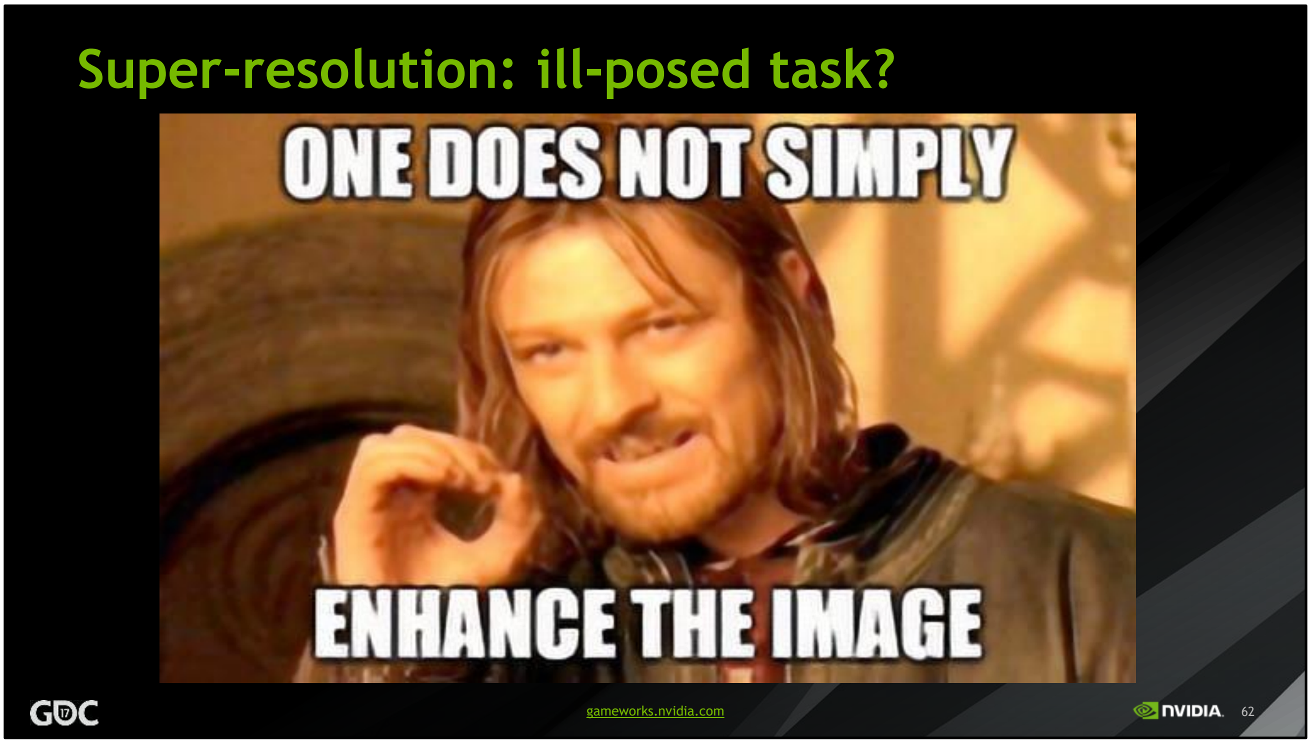
gameworks.nvidia.com



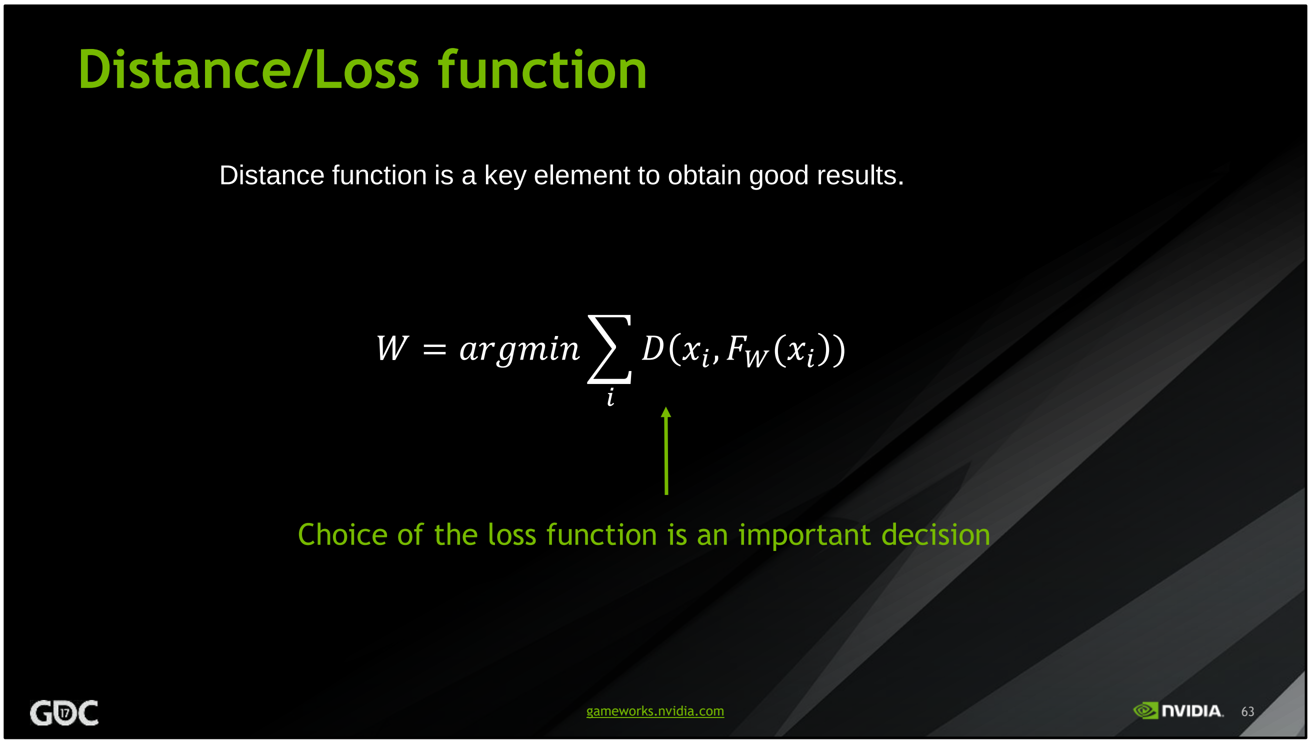
SRCAE: Inference



$$y = F_W(\hat{x})$$



Super-resolution: ill-posed task?



Distance/Loss function

Distance function is a key element to obtain good results.

$$W = argmin \sum_i D(x_i, F_W(x_i))$$

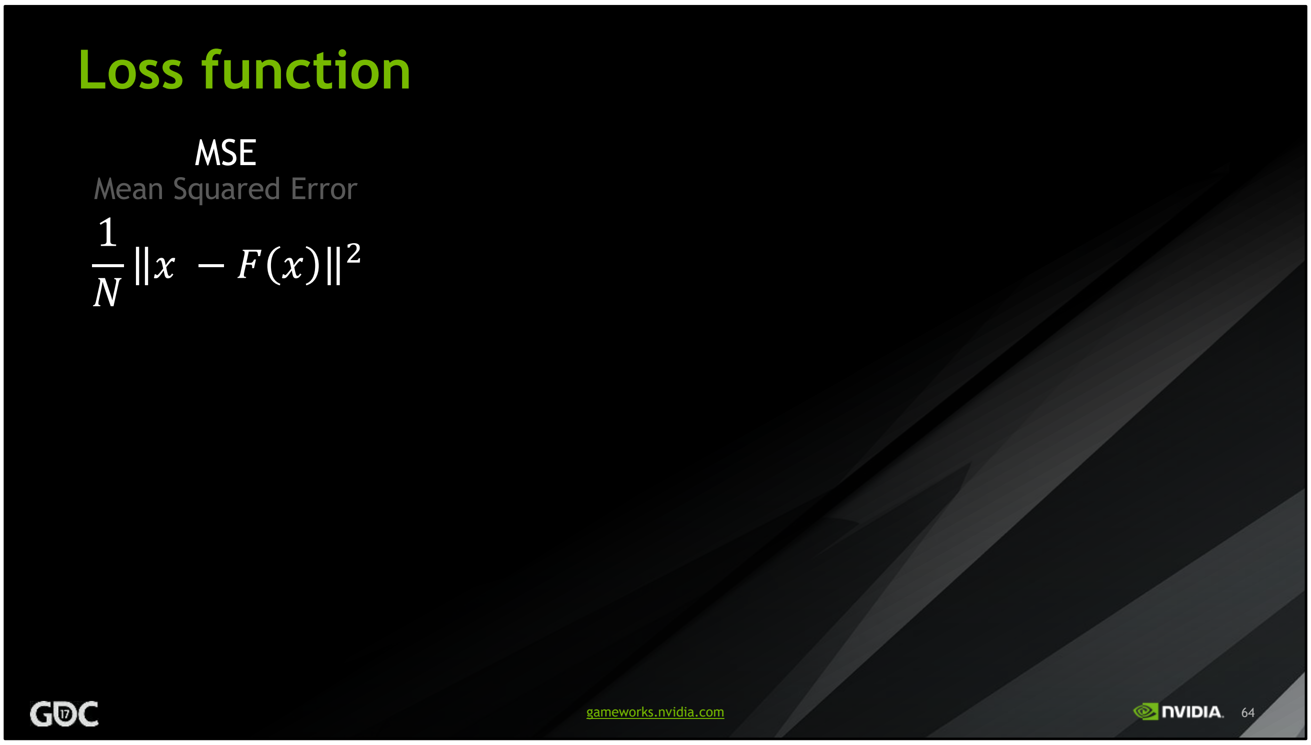
$\uparrow$

Choice of the loss function is an important decision

GDC 17 gameworks.nvidia.com NVIDIA 63

MSE, L2 and L1 metrics will eventually converge to the results shown before, and indeed when we started we was using MSE.

We started with MSE, but we obtained better results with another metric.



Loss function

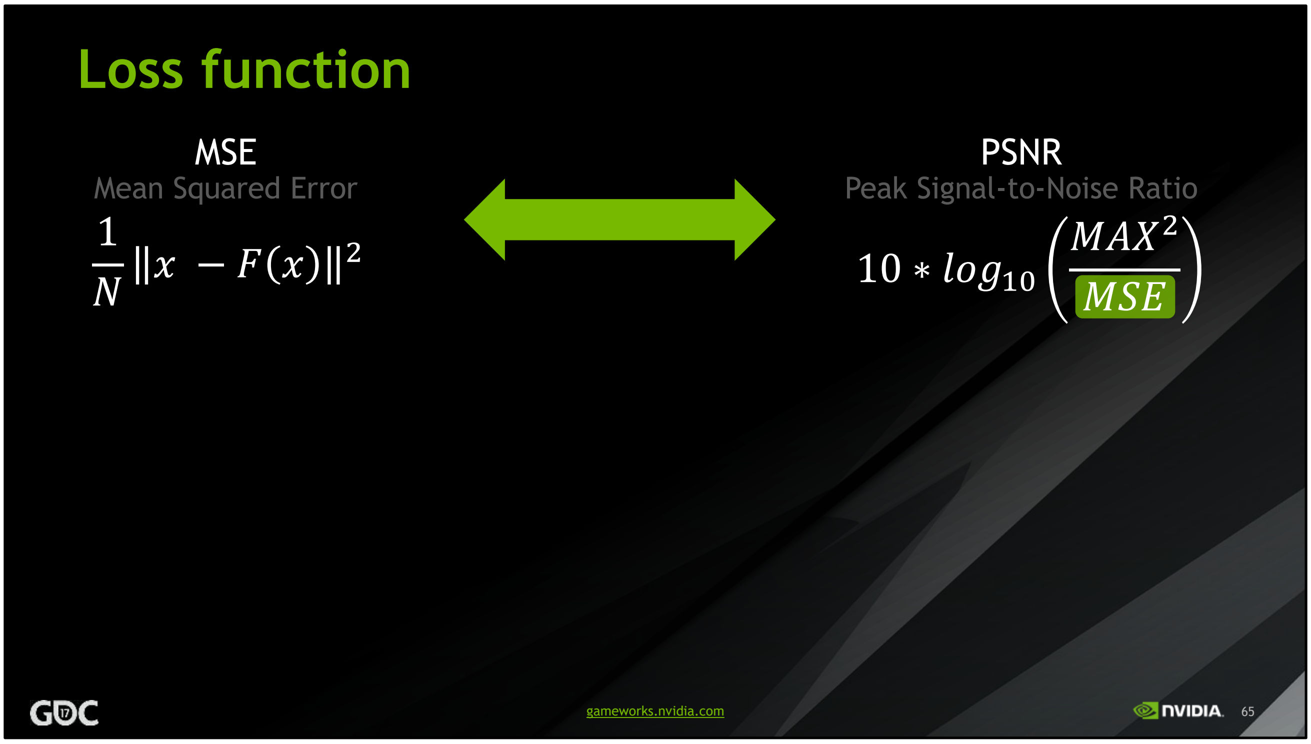
MSE

Mean Squared Error

$$\frac{1}{N}\|x - F(x)\|^2$$

GDC 17 gameworks.nvidia.com NVIDIA 64

Loss function is important. Generally, people use the MSE loss function, which stands for mean squared error.



Loss function

MSE

Mean Squared Error

$$\frac{1}{N}\|x - F(x)\|^2$$

PSNR

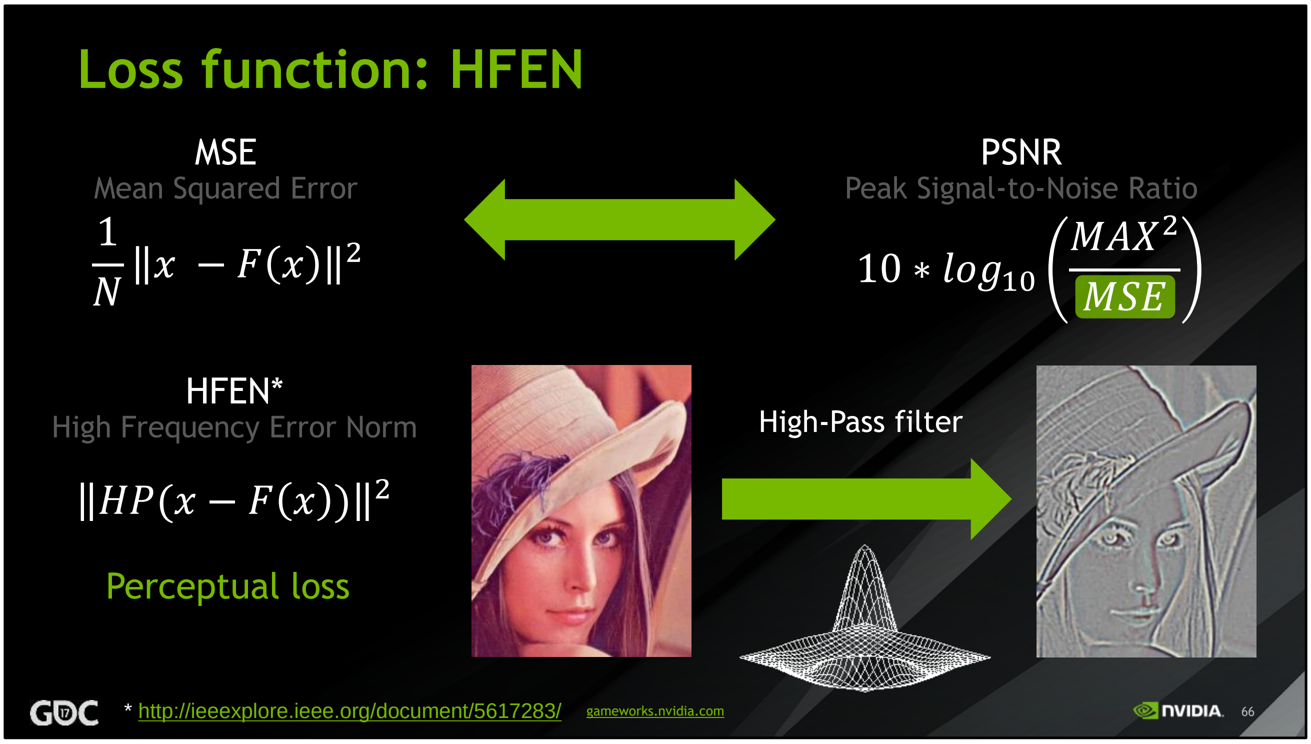
Peak Signal-to-Noise Ratio

$$10 * log_{10}\left(\frac{MAX^2}{\boxed{MSE}}\right)$$

gameworks.nvidia.com

65

Since we are solving an image reconstruction task, it is good to consider a correspondence between loss which we minimize and image quality metrics which we use to evaluate our reconstruction quality. It is easy to notice that MSE closely relates to well-known PSNR metric, which stands for peak signal to noise ratio. But MSE is too general, and PSNR poorly represents perceptual image quality. The solution is to find some other metric, which is closer to human perception.



Loss function: HFEN

MSE
Mean Squared Error

$$\frac{1}{N}\|x - F(x)\|^2$$

PSNR
Peak Signal-to-Noise Ratio

$$10 * log_{10}\left(\frac{MAX^2}{MSE}\right)$$

HFEN*
High Frequency Error Norm

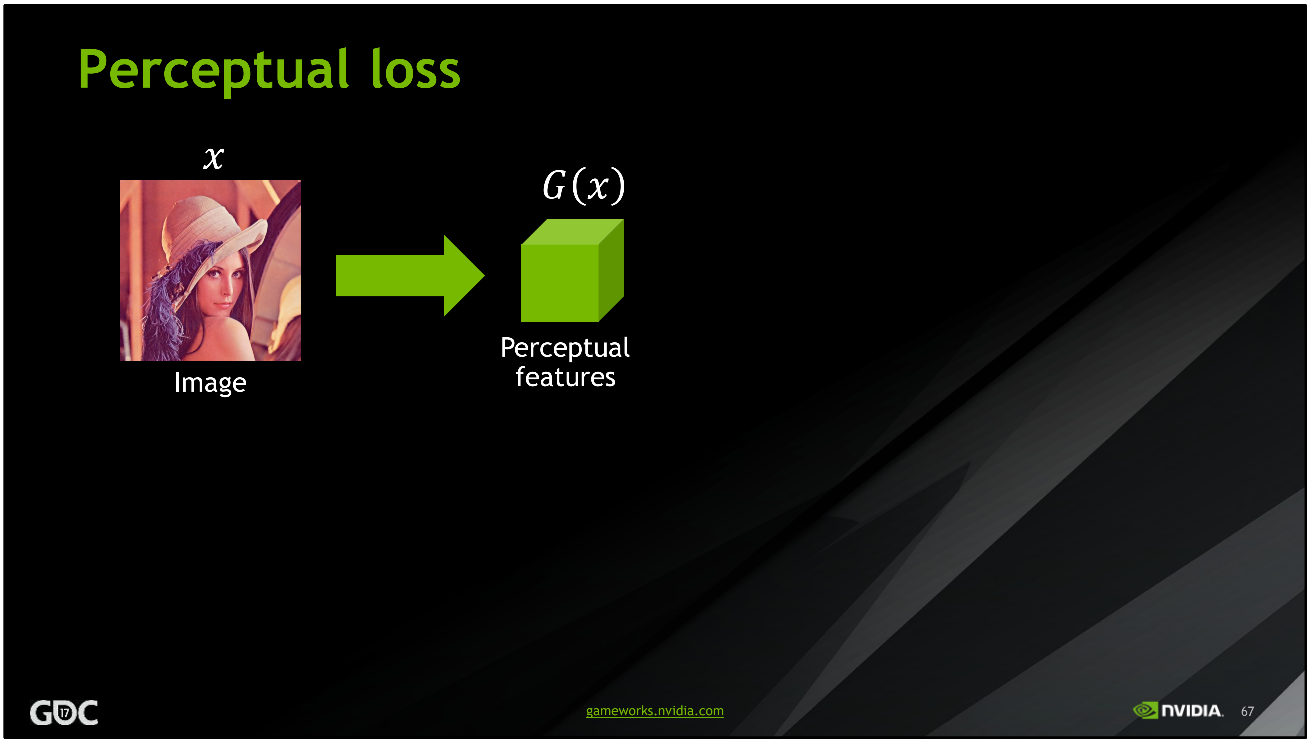
$$\|HP(x - F(x))\|^2$$

Perceptual loss

High-Pass filter

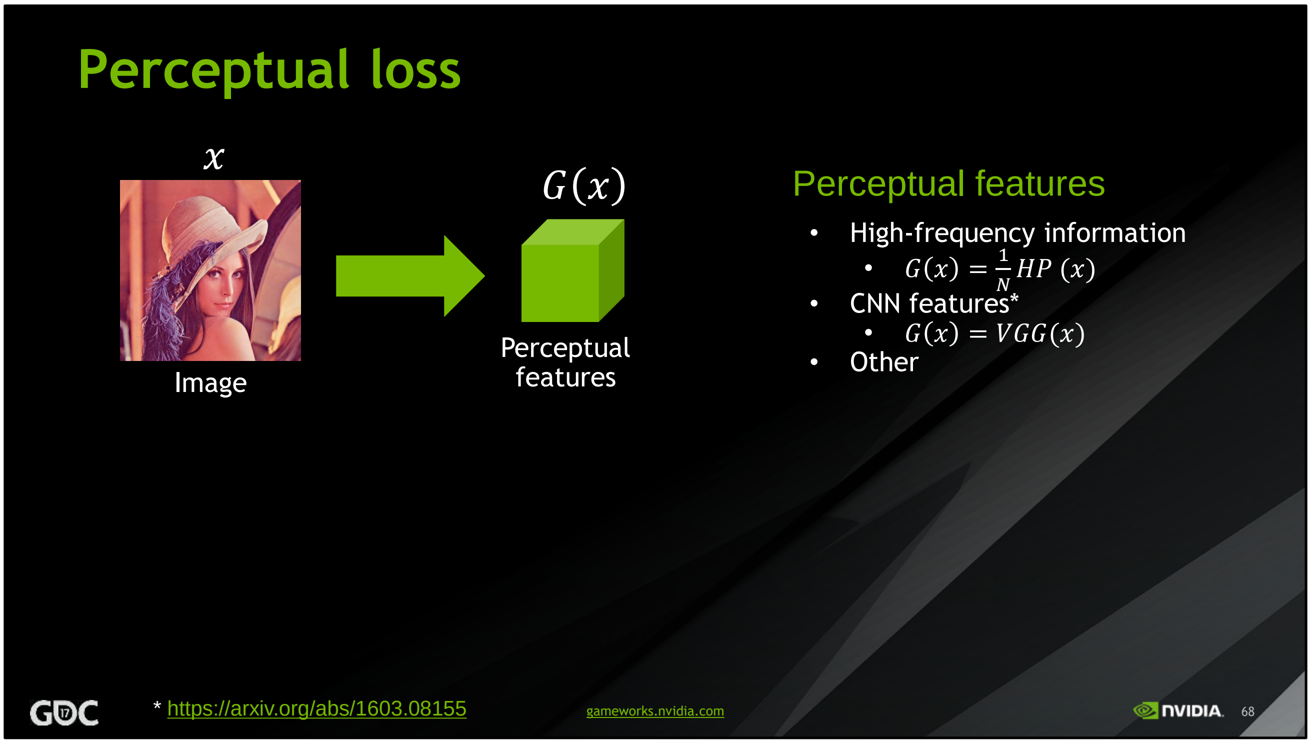
* http://ieeexplore.ieee.org/document/5617283/ gameworks.nvidia.com 66

Since we want to reconstruct fine details, a metric, which considers high-frequencies, could be useful. One of these is High Frequency Error Norm, broadly used in medical imaging. It uses High-pass operator to concentrate only on high-frequency details. Here is an example of how the operator works -- it highlights the edges. Another advantage of this operator -- it is linear, thus differentiable and easily implementable within an autoencoder loss function, which now could be considered as perceptual loss.

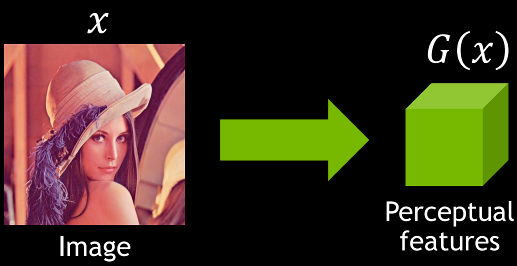


Perceptual loss

We can generalize this idea. Suppose we have some transformation, that extracts perceptual features.



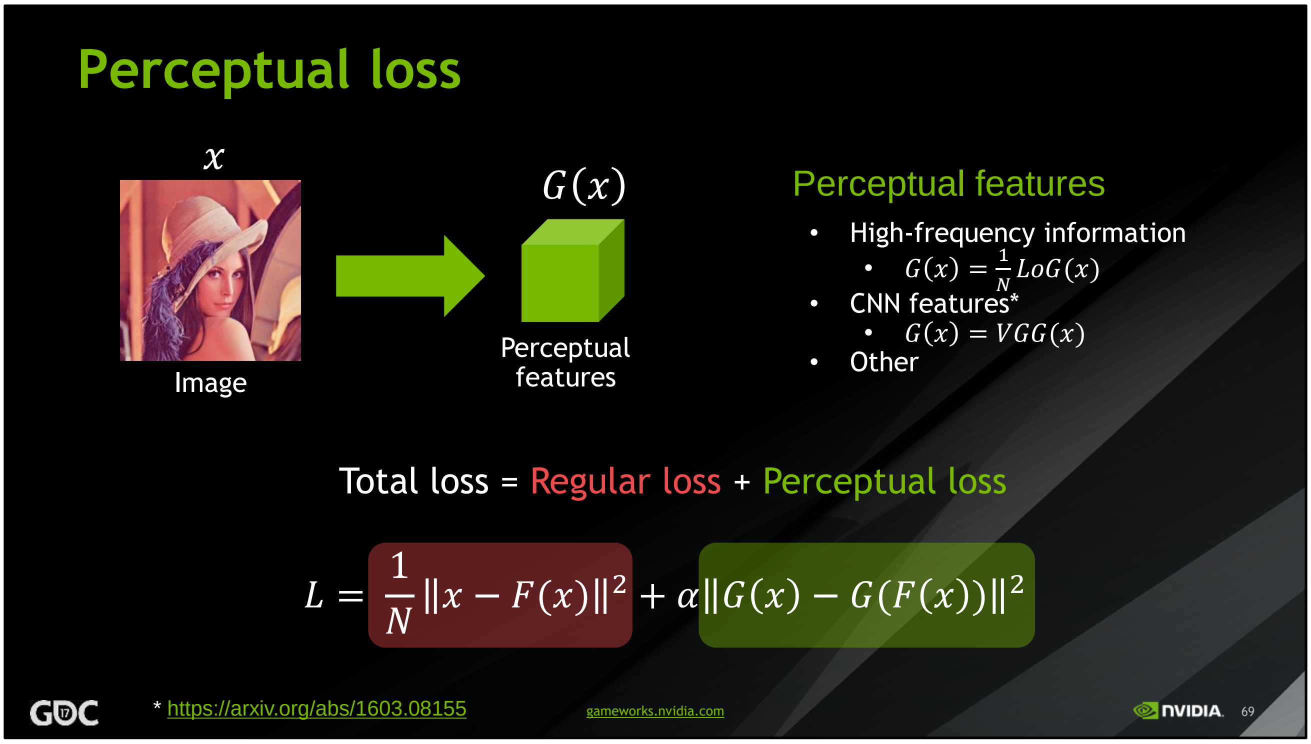
Perceptual loss



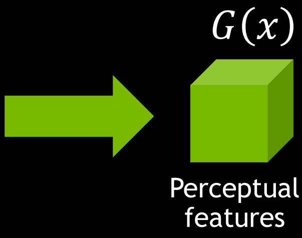
Perceptual features

- High-frequency information
  - $G(x) = \frac{1}{N} HP\ (x)$
- CNN features*
  - $G(x) = VGG(x)$
- Other

It could be the mentioned HFEN, or some other operator, extracting important details. Or it could be semantic features, extracted by means of some Convolutional Neural Network. Example -- VGG features.



Perceptual loss



Perceptual features

- High-frequency information
  - $G(x) = \frac{1}{N} LoG(x)$
- CNN features*
  - $G(x) = VGG(x)$
- Other

Total loss = Regular loss + Perceptual loss

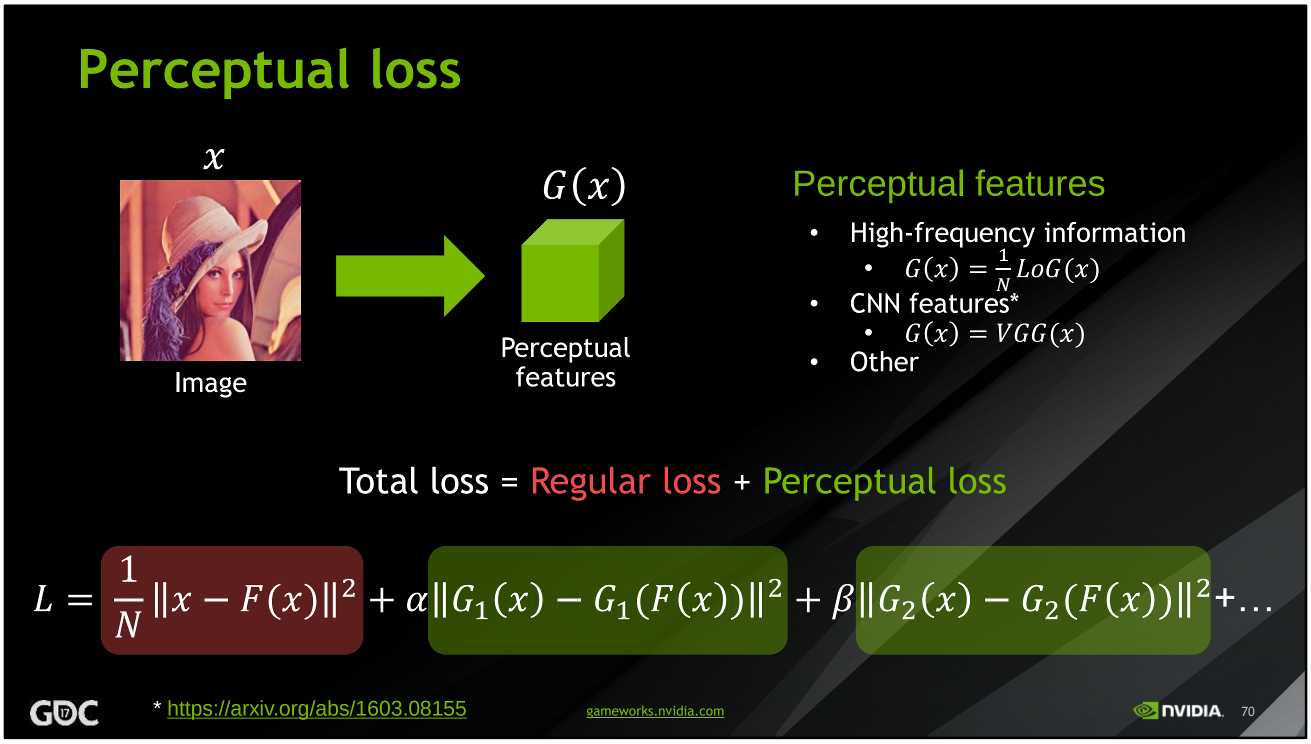
$$L = \frac{1}{N}\|x - F(x)\|^2 + \alpha\|G(x) - G(F(x))\|^2$$

* https://arxiv.org/abs/1603.08155

gameworks.nvidia.com

69

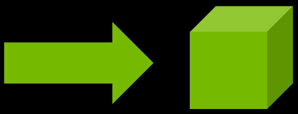
Then, having a perceptual loss, which is focused on some specific component we can construct the total loss as a weighted sum of regular content loss and the perceptual loss.



Perceptual loss

x

Image



$G(x)$

Perceptual features

Perceptual features

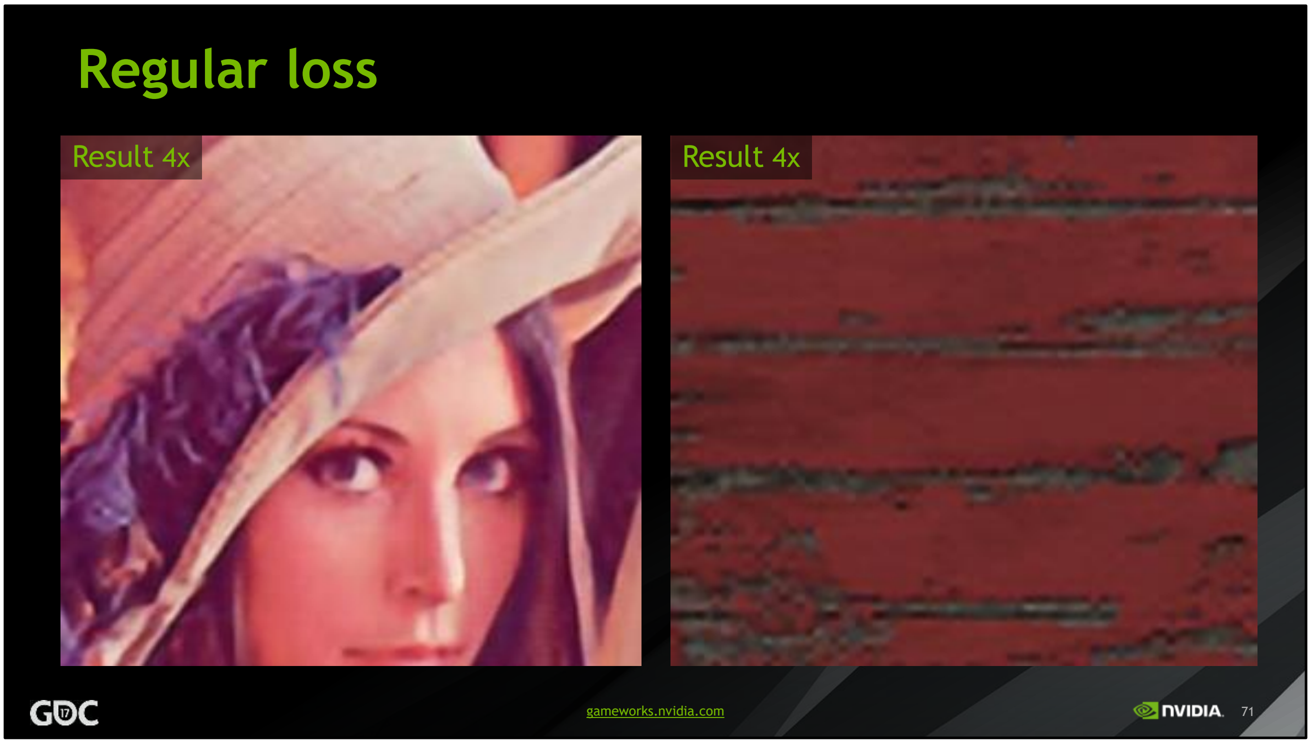
- High-frequency information
  - $G(x) = \frac{1}{N} LoG(x)$
- CNN features*
  - $G(x) = VGG(x)$
- Other

Total loss = Regular loss + Perceptual loss

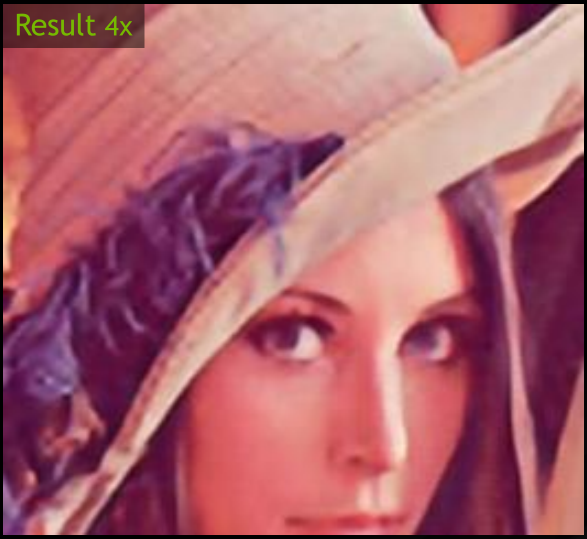
$$L = \frac{1}{N}\|x - F(x)\|^2 + \alpha\|G_1(x) - G_1(F(x))\|^2 + \beta\|G_2(x) - G_2(F(x))\|^2 + \dots$$

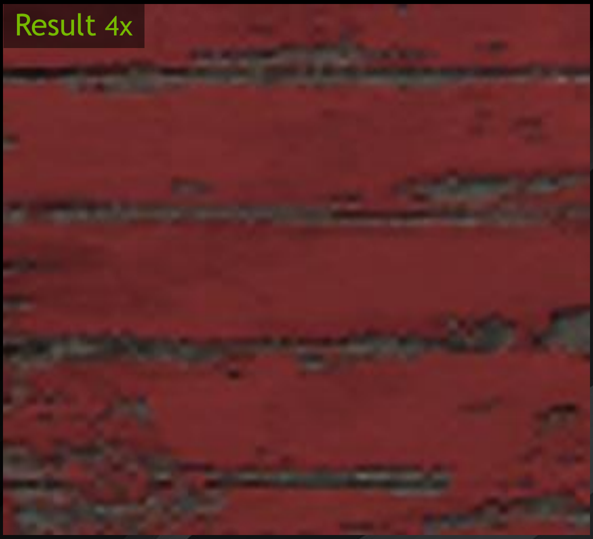
GDC * https://arxiv.org/abs/1603.08155 gameworks.nvidia.com NVIDIA 70

Or in more general case even a combination of different perceptual losses.



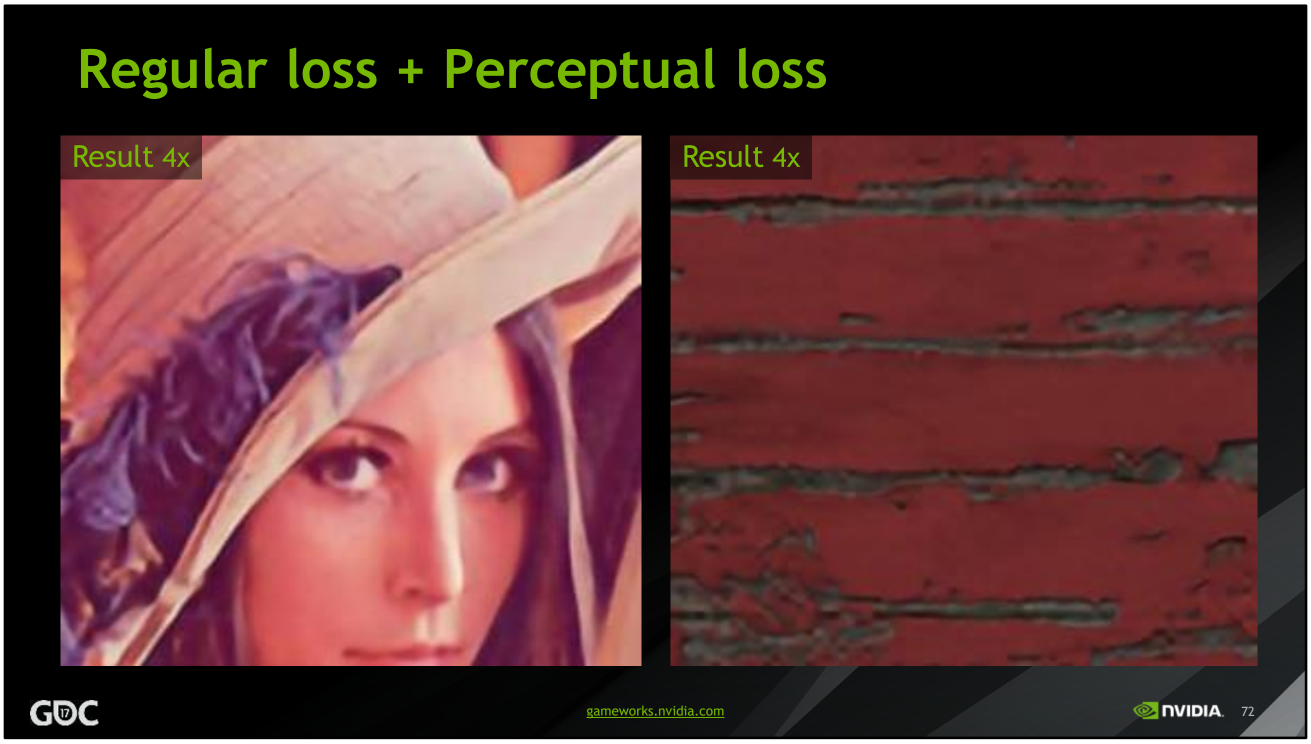
Regular loss



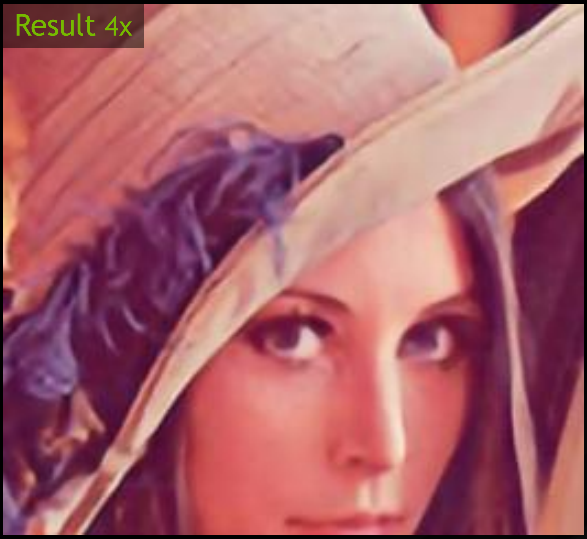


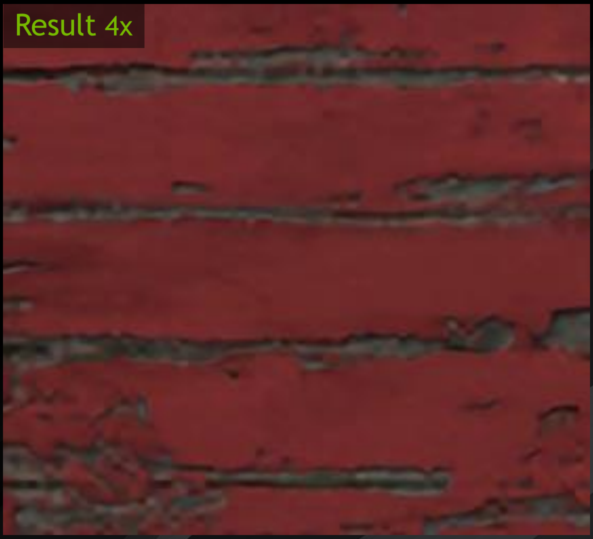
GDC 17 gameworks.nvidia.com NVIDIA 71

Here is the 4x upscaling result using regular MSE loss.



Regular loss + Perceptual loss





GDC

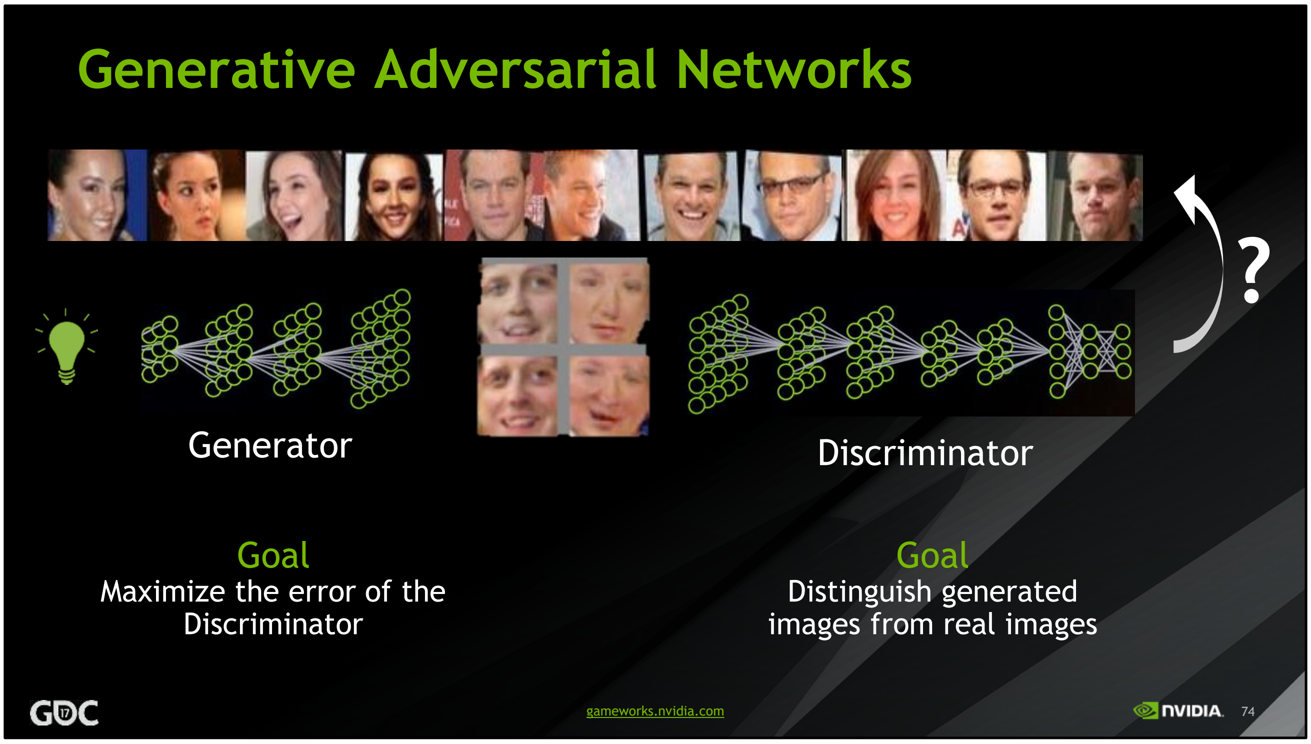
gameworks.nvidia.com

NVIDIA 72

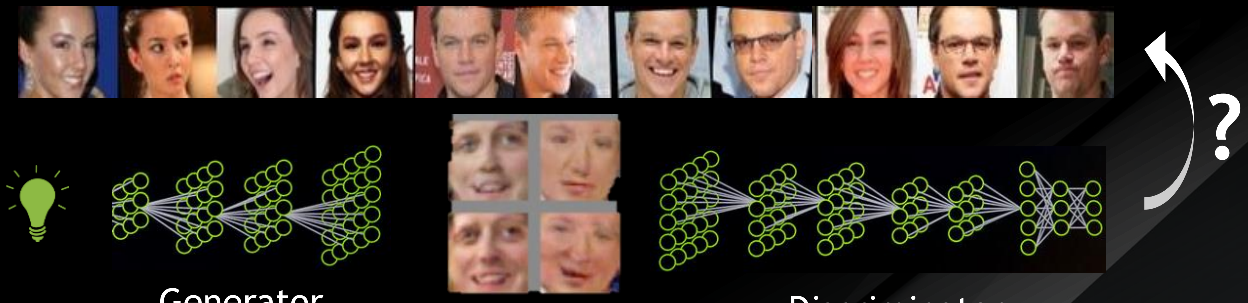
And here is the upscaling with the perceptual loss. Edges have become sharper, aliasing effect is reduced.

Demo

Super-Resolution



Generative Adversarial Networks



Generator

Discriminator

Goal
Maximize the error of the Discriminator

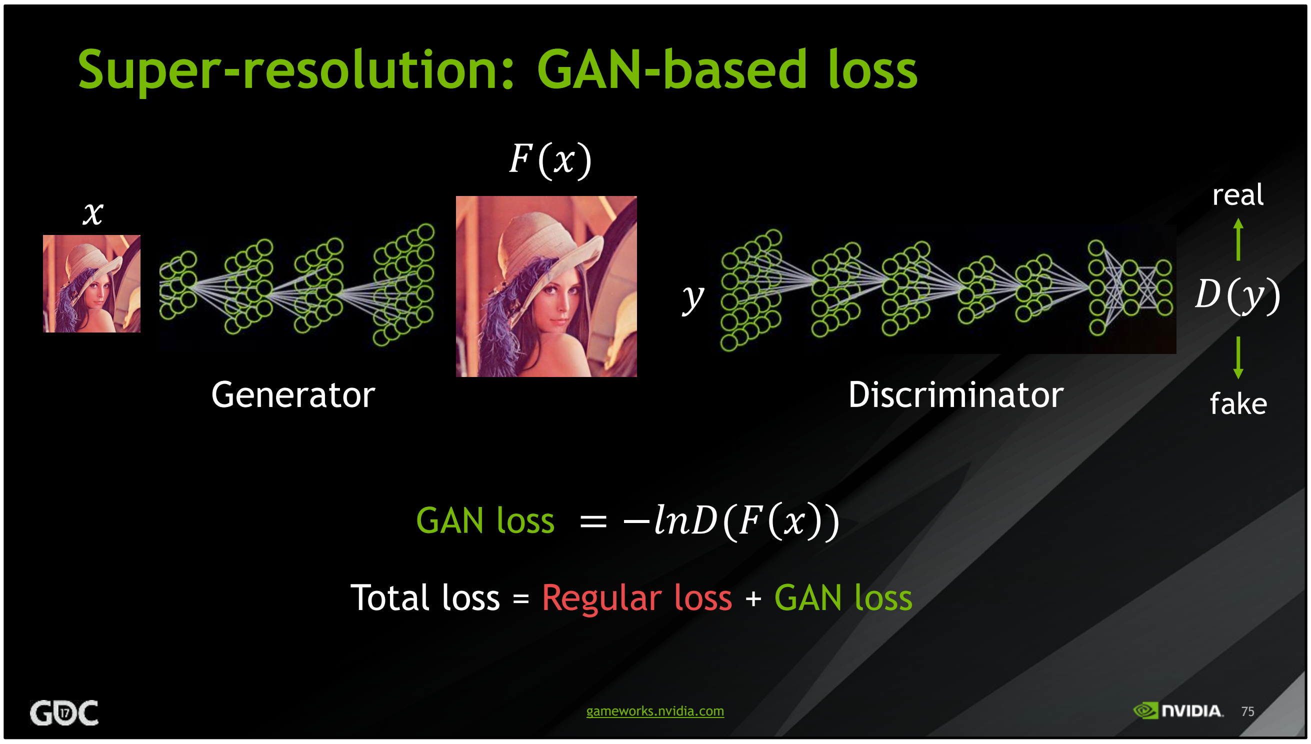
Goal
Distinguish generated images from real images

GDC

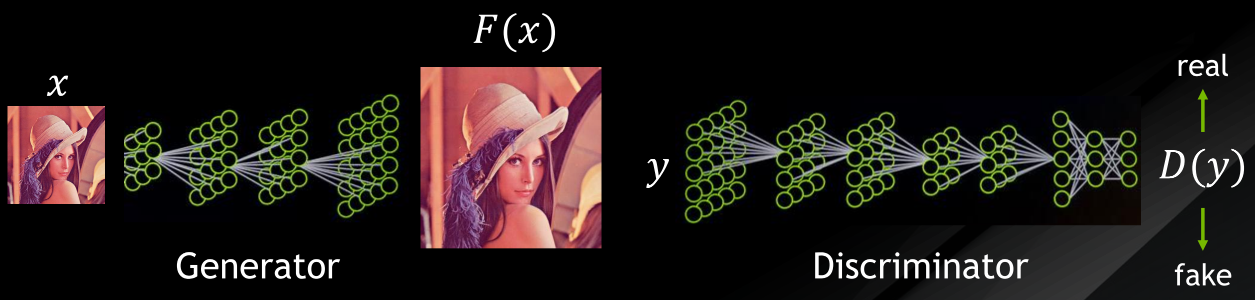
gameworks.nvidia.com

NVIDIA 74

One of the breakthrough technology in modern machine learning is Generative Adversarial Networks, or GANs. They are used to improve quality of a generative model. For example, let's say we have a generator which we want to train to generate human faces. And we want it to be good in it. For this reason, we construct a second network, called discriminator, whose goal is to distinguish generated images from the real images. Now, the goal of generator is to generate images, indistinguishable from the real ones, or similarly, maximize the error of the discriminator. They both are trained in parallel to boost their skills, and ideally we obtain a perfect generator in the end.



Super-resolution: GAN-based loss

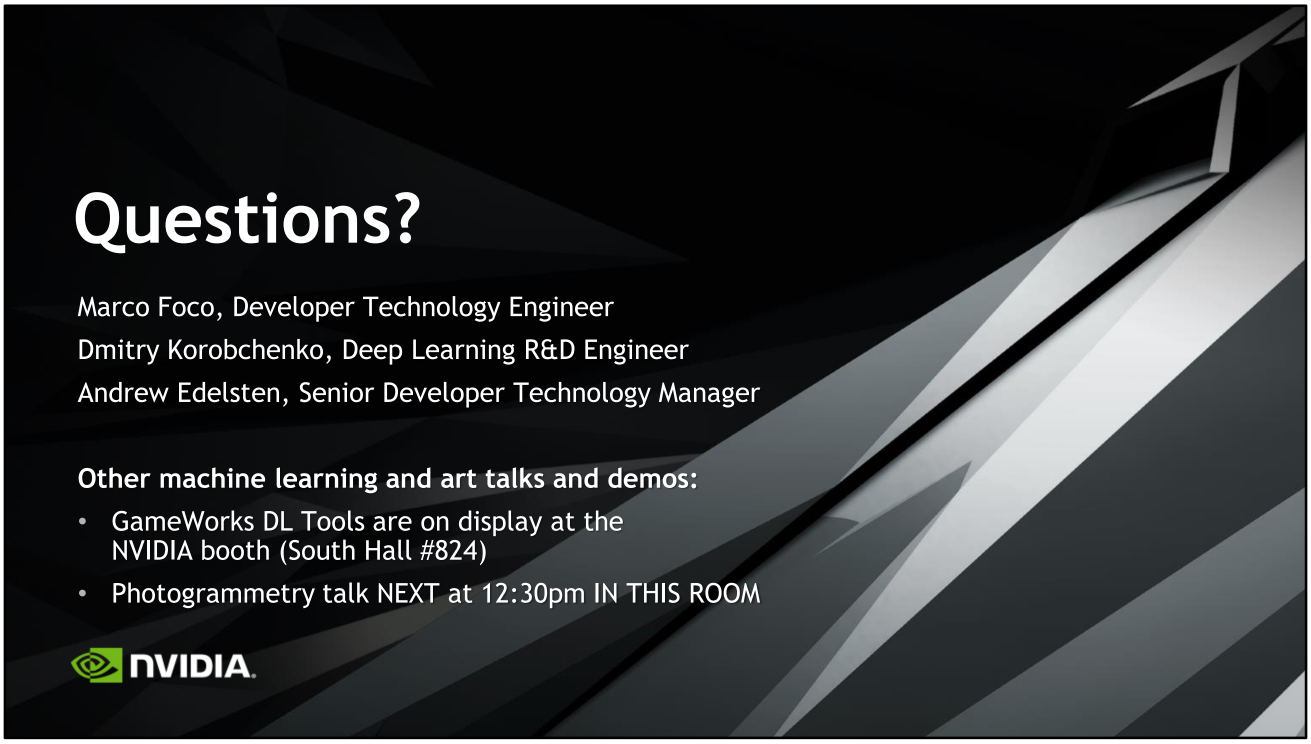


GAN loss $= -lnD(F(x))$

Total loss = Regular loss + GAN loss

Super-resolution is also a generative task. So, let's try to apply GANs to it. As a generator let's take our super-resolution auto-encoder, and as a discriminator, let's train a binary classifier, which will distinguish upscaled and real high-resolution images.

This will alter the loss function of our autoencoder, and such additional term could be considered as a special type of perceptual loss.



Questions?

Marco Foco, Developer Technology Engineer

Dmitry Korobchenko, Deep Learning R&D Engineer

Andrew Edelsten, Senior Developer Technology Manager

Other machine learning and art talks and demos:

- GameWorks DL Tools are on display at the NVIDIA booth (South Hall #824)
- Photogrammetry talk NEXT at 12:30pm IN THIS ROOM