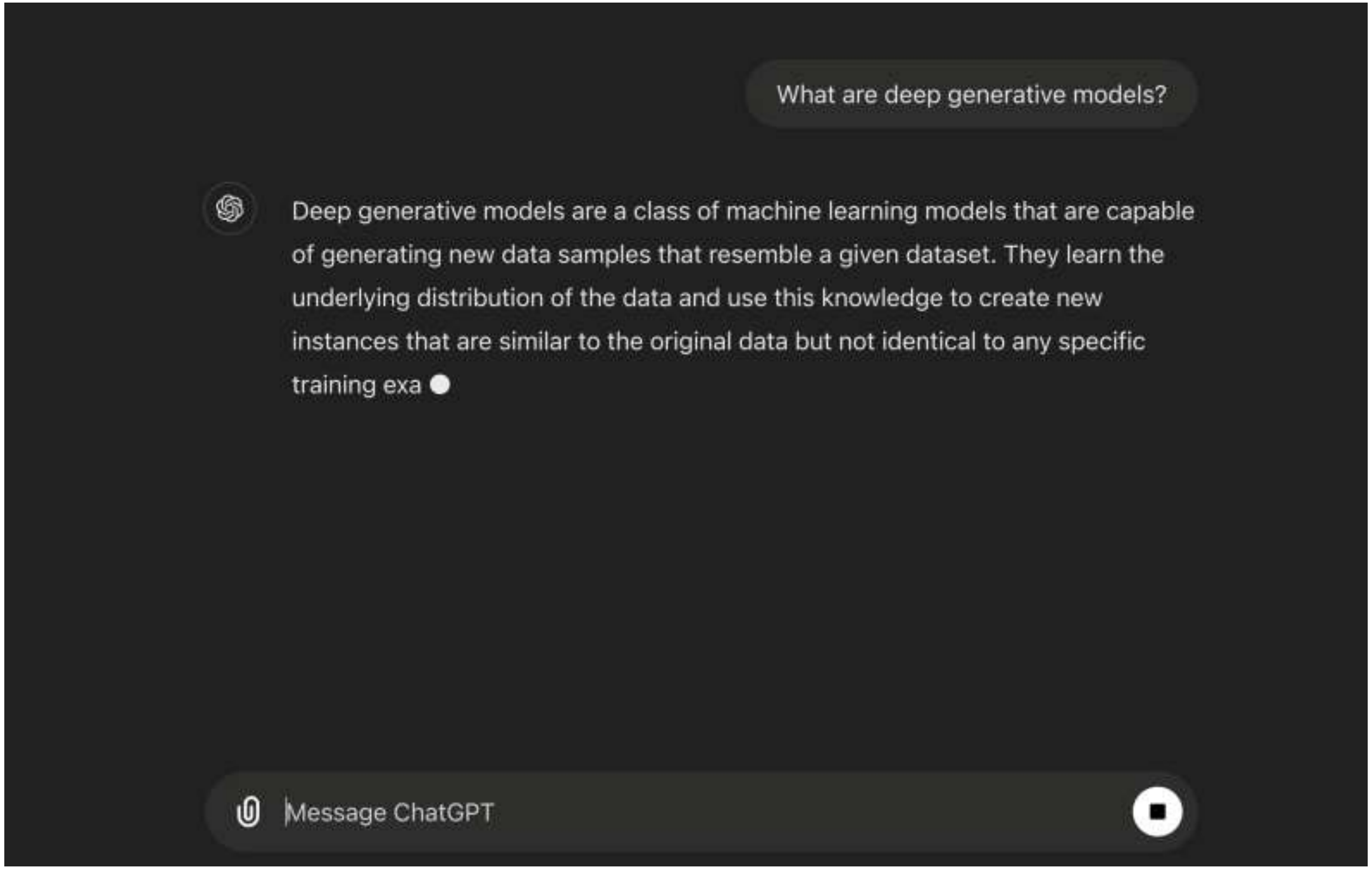
生成模型综述

The “GenAI” Era

Chatbot and natural language conversation



The “GenAI” Era

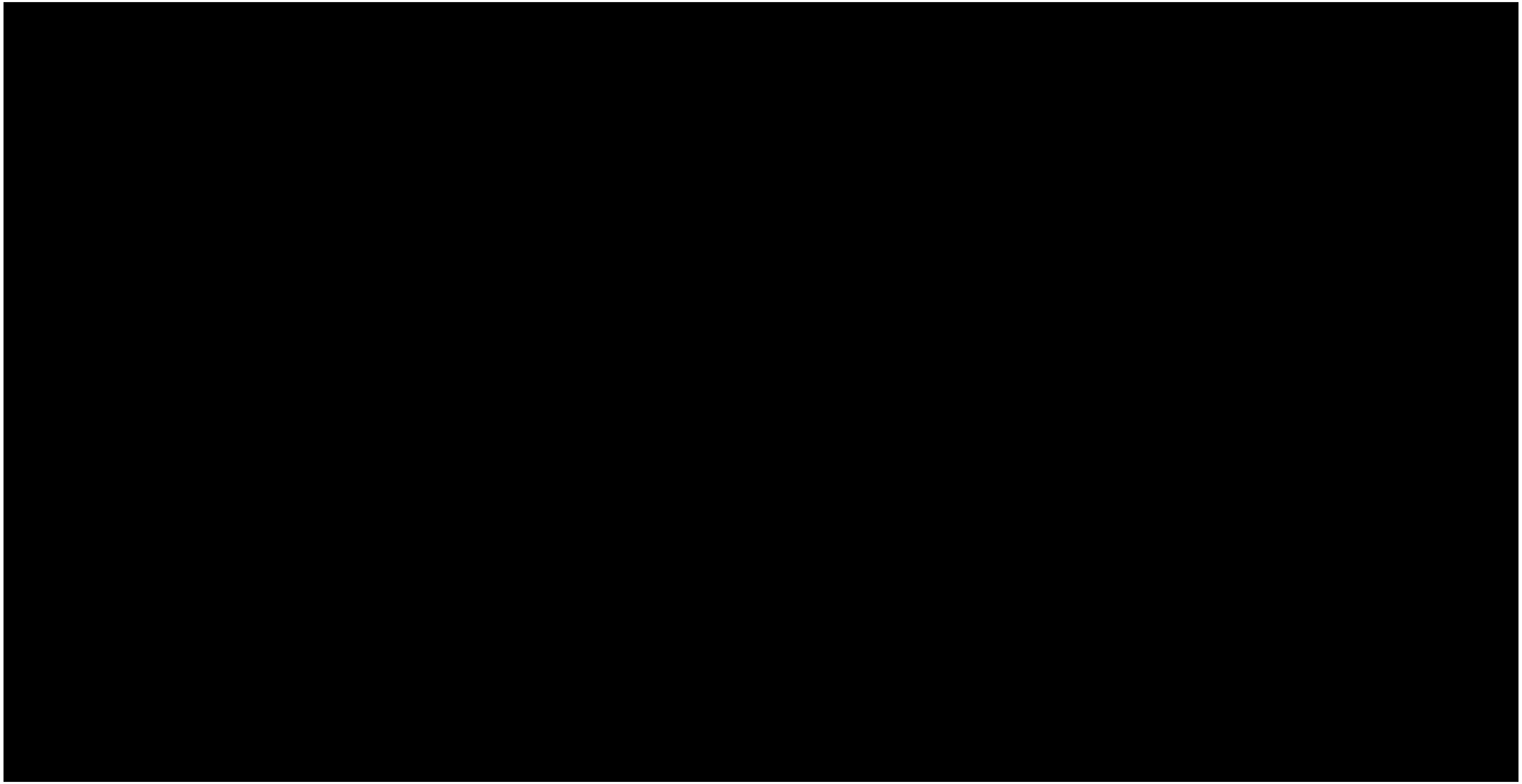
Text-to-image generation

Prompt: –*–aesthetic–*– #boho #fashion, full-body 30-something woman laying on microfloral grass, candid pose, overlay reads Stable Diffusion 3.5, cheerful cursive typography font.

https://stability.ai/news/introducing-stable-diffusion-3-5

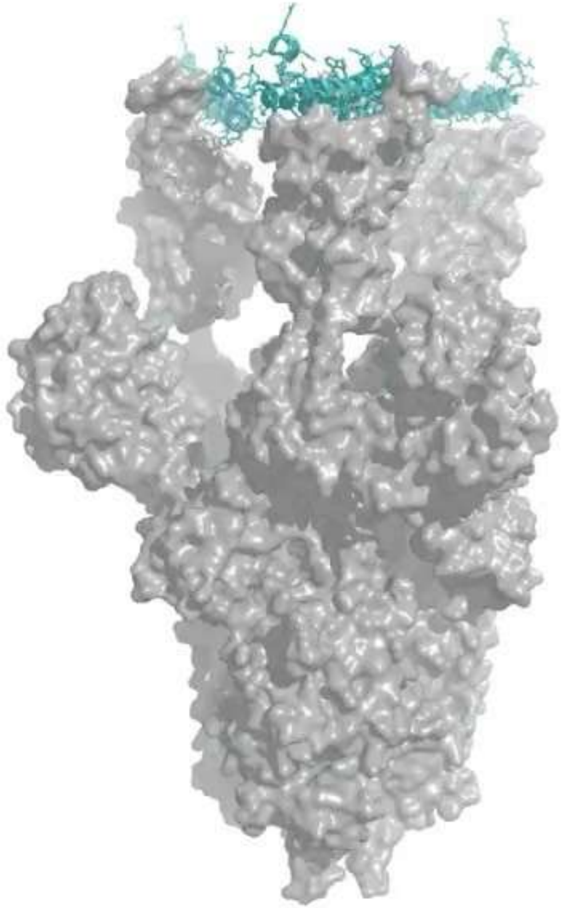
The “GenAI” Era

AI assistant for code generation



The "GenAI" Era

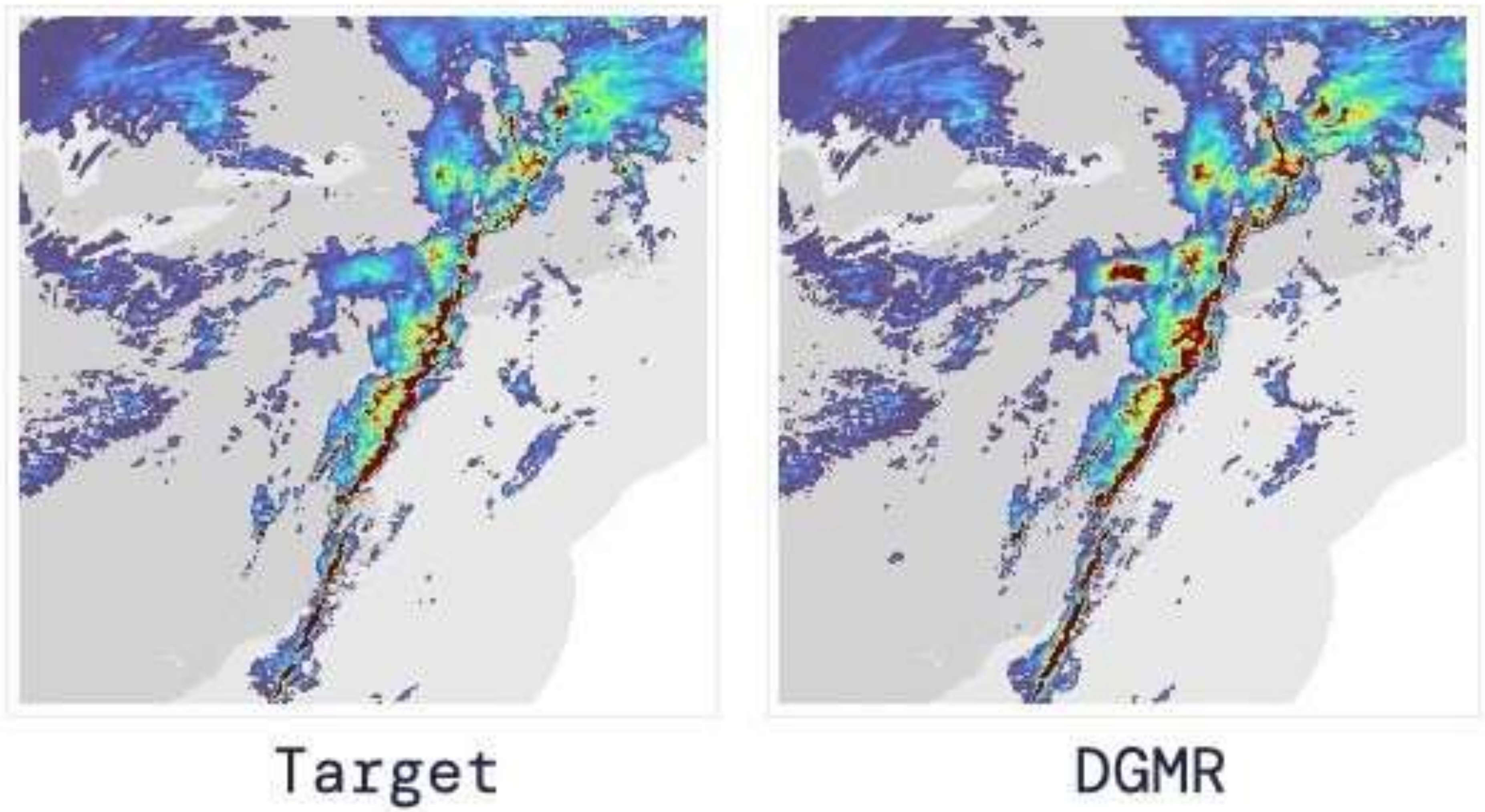
Protein design and generation



Watson, et al. De novo design of protein structure and function with RFdiffusion, Nature 2023

The “GenAI” Era

Weather forecasting



Skilful precipitation nowcasting using deep generative models of radar, Nature 2021

The “GenAI” Era

Image-to-image generation


GPT-Image-1 (GPT 4o)

The “GenAI” Era

Image-to-image generation

Nano Banana Pro

The "GenAI" Era

Image-to-image generation

Nano Banana Pro

The “GenAI” Era

Image-to-image generation

Nano Banana Pro

The “GenAI” Era

Image-to-image generation

Nano Banana Pro

The "GenAI" Era

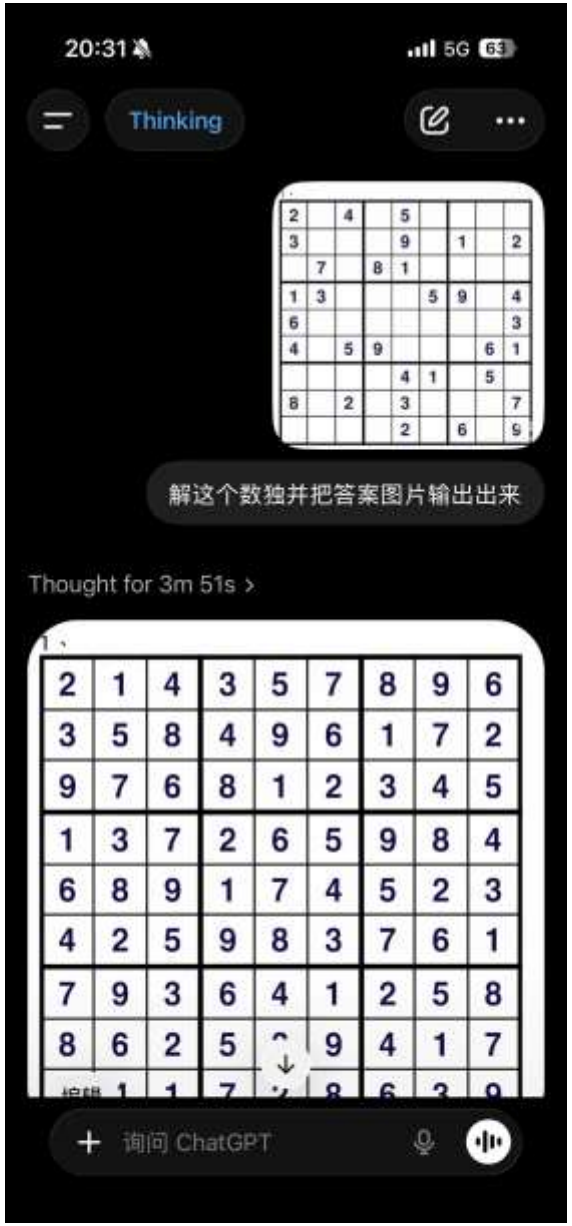
Image-to-image generation

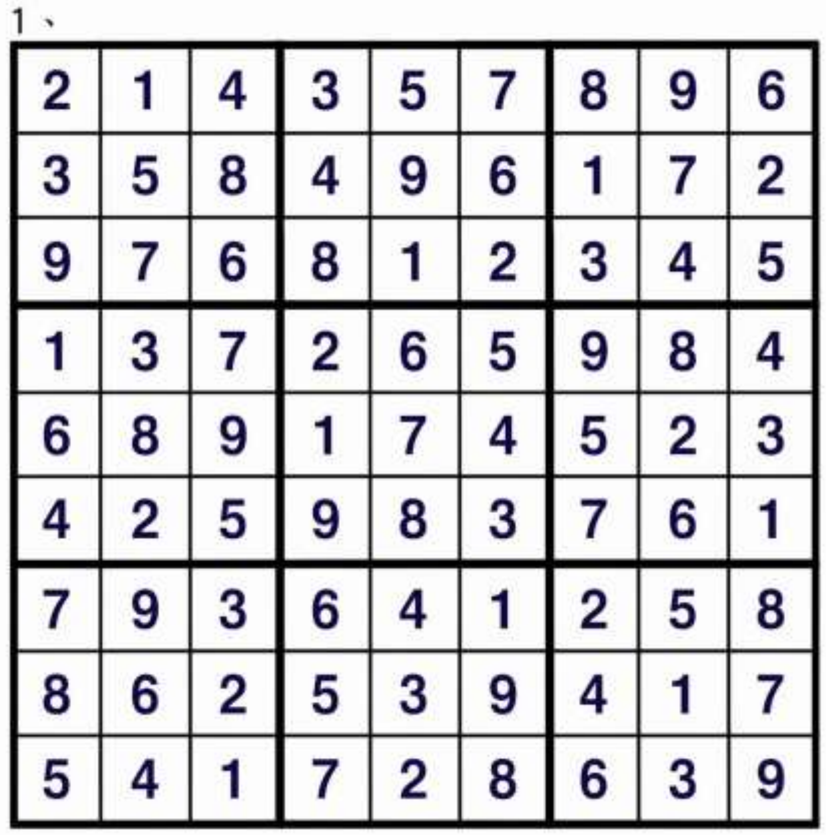
Nano Banana Pro

The “GenAI” Era

GPT-Image-2

The “GenAI” Era





GPT-Image-2

The “GenAI” Era

GPT-Image-2

The “GenAI” Era

学校 OpenAI实验学校 姓名 ChatGPT 班 九年级1班 学号 20260422

北京市海淀区九年级第二学期期末练习

物 理

2024.6

考生须知	1. 本试卷共8页，共五道大题，28道小题，满分100分。考试时间90分钟。 2. 在试卷和答题卡上准确填写学校名称、班级名称、姓名和考号。 3. 试题答案一律填涂或书写在答题卡上，在试卷上作答无效。 4. 在答题卡上，选择题用2B铅笔作答，其他试题用黑色字迹签字笔作答。 5. 考试结束，请将本试卷和答题卡一并交回。

一、单项选择题（下列各小题均有四个选项，其中只有一个选项符合题意。共24分，每小题2分）

1. 图1所示的实例中，目的是减小摩擦的是

行李箱安装滚动轮 A　自行车把上装有橡胶套 B　鞋底刻有凹凸花纹 C　滑雪时用雪橇板滑行 D

图1

滚动代替滑动，减小摩擦

2. 图2所示的工具中，在使用时属于费力杠杆的是

A. 羊角锤　B. 钢丝钳　C. 食品夹　D. 羊角扳手

图2 食品夹费力杠杆，省距离

3. 图3所示的实例中，属于增大压强的是

铁轨铺在枕木上　书包带做得较宽　载重车装有多个车轮　D 刀刃磨得很锋利

图3 刀刃锋利，受力面积小，压强大

九年级物理期末练习 第1页（共8页）

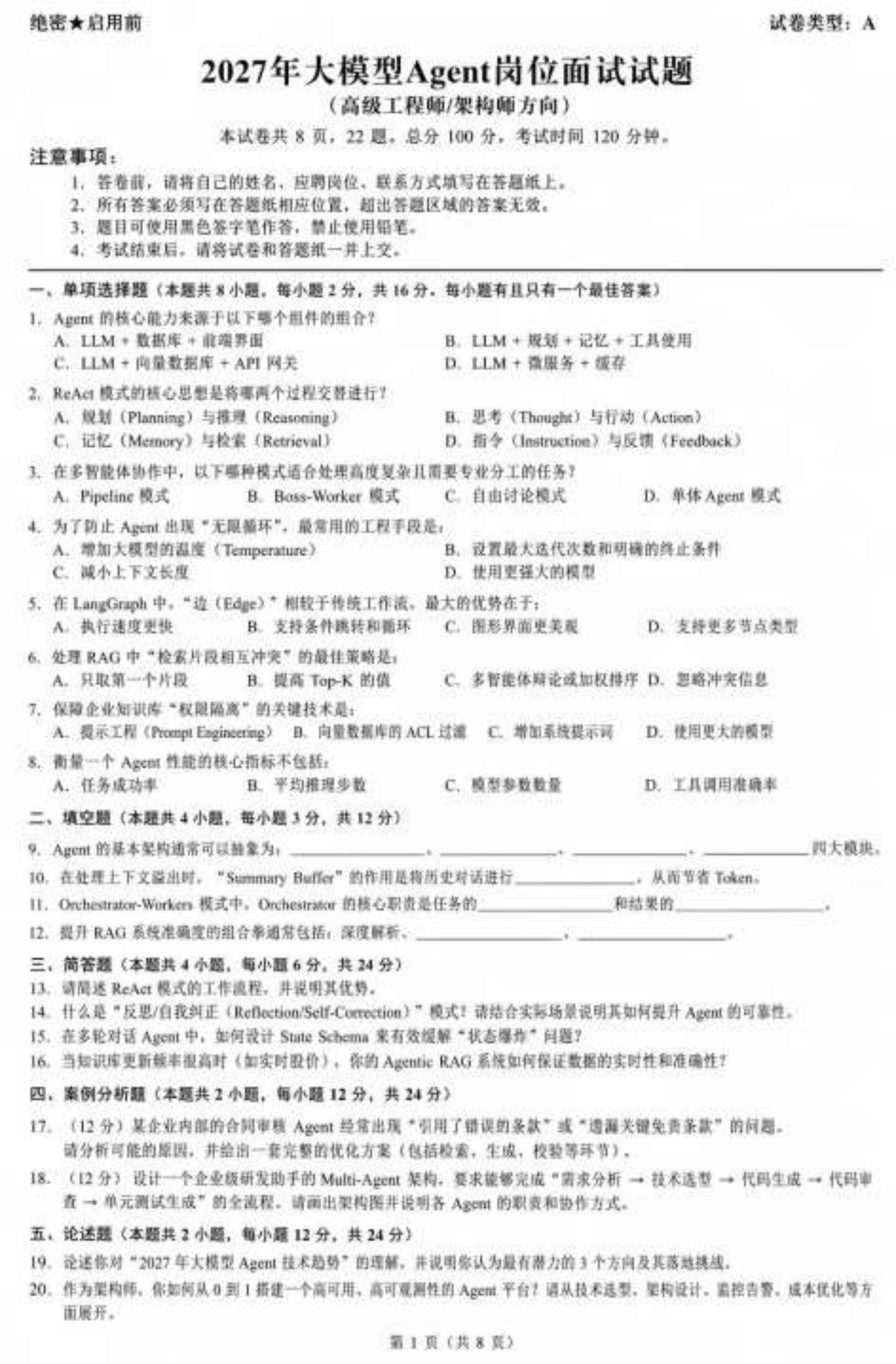
GPT-Image-2

The “GenAI” Era

GPT-Image-2

The “GenAI” Era

GPT-Image-2



绝密★启用前　　试卷类型：A

2027年大模型Agent岗位面试试题

（高级工程师/架构师方向）

本试卷共 8 页，22 题，总分 100 分，考试时间 120 分钟。

注意事项：

1. 答卷前，请将自己的姓名、应聘岗位、联系方式填写在答题纸上。
2. 所有答案必须写在答题纸相应位置，超出答题区域的答案无效。
3. 题目可使用黑色签字笔作答，禁止使用铅笔。
4. 考试结束后，请将试卷和答题纸一并上交。

一、单项选择题（本题共 8 小题，每小题 2 分，共 16 分，每小题有且只有一个最佳答案）

1. Agent 的核心能力来源于以下哪个组件的组合？
A. LLM + 数据库 + 前端界面　B. LLM + 规划 + 记忆 + 工具使用
C. LLM + 向量数据库 + API 网关　D. LLM + 微服务 + 缓存

2. ReAct 模式的核心思想是将哪两个过程交替进行？
A. 规划（Planning）与推理（Reasoning）　B. 思考（Thought）与行动（Action）
C. 记忆（Memory）与检索（Retrieval）　D. 指令（Instruction）与反馈（Feedback）

3. 在多智能体协作中，以下哪种模式适合处理高度复杂且需要专业分工的任务？
A. Pipeline 模式　B. Boss-Worker 模式　C. 自由讨论模式　D. 单体 Agent 模式

4. 为了防止 Agent 出现“无限循环”，最常用的工程手段是：
A. 增加大模型的温度（Temperature）　B. 设置最大迭代次数和明确的终止条件
C. 减小上下文长度　D. 使用更强大的模型

5. 在 LangGraph 中，“边（Edge）”相较于传统工作流，最大的优势在于：
A. 执行速度更快　B. 支持条件跳转和循环　C. 图形界面更美观　D. 支持更多节点类型

6. 处理 RAG 中“检索片段相互冲突”的最佳策略是：
A. 只取第一个片段　B. 提高 Top-K 的值　C. 多智能体辩论或加权排序　D. 忽略冲突信息

7. 保障企业知识库“权限隔离”的关键技术是：
A. 提示工程（Prompt Engineering）　B. 向量数据库的 ACL 过滤　C. 增加系统提示词　D. 使用更大的模型

8. 衡量一个 Agent 性能的核心指标不包括：
A. 任务成功率　B. 平均推理步数　C. 模型参数数量　D. 工具调用准确率

二、填空题（本题共 4 小题，每小题 3 分，共 12 分）

9. Agent 的基本架构通常可以抽象为：________、________、________、________四大模块。
10. 在处理上下文溢出时，“Summary Buffer”的作用是将历史对话进行________，从而节省 Token。
11. Orchestrator-Workers 模式中，Orchestrator 的核心职责是任务的________和结果的________。
12. 提升 RAG 系统准确度的组合拳通常包括：深度解析、________、________。

三、简答题（本题共 4 小题，每小题 6 分，共 24 分）

13. 请简述 ReAct 模式的工作流程，并说明其优势。
14. 什么是“反思/自我纠正（Reflection/Self-Correction）”模式？请结合实际场景说明其如何提升 Agent 的可靠性。
15. 在多轮对话 Agent 中，如何设计 State Schema 来有效缓解“状态爆炸”问题？
16. 当知识库更新频率很高时（如实时股价），你的 Agentic RAG 系统如何保证数据的实时性和准确性？

四、案例分析题（本题共 2 小题，每小题 12 分，共 24 分）

17. （12 分）某企业内部的合同审核 Agent 经常出现“引用了错误的条款”或“遗漏关键免责条款”的问题。请分析可能的原因，并给出一套完整的优化方案（包括检索、生成、校验等环节）。
18. （12 分）设计一个企业级研发助手的 Multi-Agent 架构，要求能够完成“需求分析 → 技术选型 → 代码生成 → 代码审查 → 单元测试生成”的全流程。请画出架构图并说明各 Agent 的职责和协作方式。

五、论述题（本题共 2 小题，每小题 12 分，共 24 分）

19. 论述你对“2027 年大模型 Agent 技术趋势”的理解，并说明你认为最有潜力的 3 个方向及其落地挑战。
20. 作为架构师，你如何从 0 到 1 搭建一个高可用、高可观测性的 Agent 平台？请从技术选型、架构设计、监控告警、成本优化等方面展开。

第 1 页（共 8 页）

The “GenAI” Era

GPT-Image-2

绝密★启用前 姓名：ChatGPT 应聘岗位：Agent架构师 联系方式：chatgpt@openai.com 试卷类型：A

2027年大模型Agent岗位面试试题

（高级工程师/架构师方向）

本试卷共 8 页，22 题，总分 100 分，考试时间 120 分钟。

注意事项：

1. 答卷前，请将自己的姓名、应聘岗位、联系方式填写在答题纸上。
2. 所有答案必须写在答题纸相应位置，超出答题区域的答案无效。
3. 题目可使用黑色签字笔作答，禁止使用铅笔。
4. 考试结束后，请将试卷和答题纸一并上交。

一、单项选择题（本题共 8 小题，每小题 2 分，共 16 分。每小题有且只有一个最佳答案）

1. B 1. Agent 的核心能力来源于以下哪个组件的组合？
A. LLM + 数据库 + 前端界面 B. LLM + 规划 + 记忆 + 工具使用
C. LLM + 向量数据库 + API 网关 D. LLM + 微服务 + 缓存

2. B 2. ReAct 模式的核心思想是将哪两个过程交替进行？
A. 规划（Planning）与推理（Reasoning） B. 思考（Thought）与行动（Action）
C. 记忆（Memory）与检索（Retrieval） D. 指令（Instruction）与反馈（Feedback）

3. B 3. 在多智能体协作中，以下哪种模式适合处理高度复杂且需要专业分工的任务？
A. Pipeline 模式 B. Boss-Worker 模式 C. 自由讨论模式 D. 单体 Agent 模式

4. B 4. 为了防止 Agent 出现“无限循环”，最常用的工程手段是：
A. 增加大模型的温度（Temperature） B. 设置最大迭代次数和明确的终止条件
C. 减小上下文长度 D. 使用更强大的模型

5. B 5. 在 LangGraph 中，“边（Edge）”相较于传统工作流，最大的优势在于：
A. 执行速度更快 B. 支持条件跳转和循环 C. 图形界面更美观 D. 支持更多节点类型

6. C 7. 处理 RAG 中“检索片段相互冲突”的最佳策略是：
A. 只取第一个片段 B. 提高 Top-K 的值 C. 多智能体辩论或加权排序 D. 忽略冲突信息

8. B 8. 衡量一个 Agent 性能的核心指标不包括：
A. 任务成功率 B. 平均推理步数 C. 模型参数数量 D. 工具调用准确率

二、填空题（本题共 4 小题，每小题 3 分，共 12 分）

9. Agent 的基本架构通常可以抽象为：规划（Planning）、记忆（Memory）、工具使用（Tool Use）、行动（Action）四大模块。
10. 在处理上下文溢出时，“Summary Buffer”的作用是将历史对话进行摘要压缩/总结归纳，从而节省 Token。
11. Orchestrator-Workers 模式中，Orchestrator 的核心职责是任务的拆解分发和结果的结果的汇总整合。
12. 提升 RAG 系统准确度的组合拳通常包括：深度解析、混合检索、重排序（Re-ranking）。

三、简答题（本题共 4 小题，每小题 6 分，共 24 分）

13. 请简述 ReAct 模式的工作流程，并说明其优势。ReAct：Thought → Action → Observation 循环；先思考分解任务,再调用执行,
依据 Observation 修正后续推理，直到终止。优点：减少幻觉，提升复杂任务完成率，可解释性强。

14. 什么是“反思/自我纠正（Reflection/Self-Correction）”模式？请结合实际场景说明其如何提升 Agent 的可靠性。
Reflection / Self-Correction：先生成初稿，再用规则/模型自检，发现错误后重写。
例：合同审查先给结论，再核对引用条款与页码，不一致则回查知识库并修正。可降低误引、漏检。

15. 在多轮对话 Agent 中，如何设计 State Schema 来有效缓解“状态爆炸”问题？
State Schema：状态分层结构化（用户信息/目标/计划/工具结果/长期记忆））；仅保留关键字段，
使用 summary buffer、TTL、版本号和事件驱动更新，避免状态爆炸。

16. 当知识库更新频率很高时（如实时股价），你的 Agentic RAG 系统如何保证数据的实时性和准确性？
实时 RAG：增量索引 + 事件驱动刷新；结果带时间戳/版本；优先回源实时库；混合检索 + rerank；
缓存失效与答案校验并行，保证时效与准确。

四、案例分析题（本题共 2 小题，每小题 12 分，共 24 分）

17. （12 分）某企业内部的合同审核 Agent 经常出现“引用了错误的条款”或“遗漏关键免责条款”的问题。
请分析可能的原因，并给出一套完整的优化方案（包括检索、生成、校验等环节）。
原因：①知识库版本混杂；②chunk 切分不合理；③仅语义召回，未按合同类型/法域/生效时间过滤；
④生成未强制“结论-依据-页码”对齐；⑤缺少规则校验。
方案：检索端做元数据过滤 + Hybrid Search + Rerank；生成端结构化输出{风险点、引用条款、原文摘录、
页码、建议}；校验端用规则引擎核对条款编号、免责条款清单、是否过期；高风险 case 人工复核。

18. （12 分）设计一个企业级研发助手的 Multi-Agent 架构，要求能够完成“需求分析 → 技术选型 → 代码生成 → 代码审查 → 单元测试生成”的全流程。请画出架构图并说明各 Agent 的职责和协作方式。
多 Agent：Coordinator/PM Agent → Architect Agent → Coder Agent → Reviewer Agent → Test Agent，
职责分别是需求解析与调度、技术选型与架构设计、代码生成、代码审查、安全与规范检查、单测/
集成测试生成与执行。协作链路：需求分析 → 技术选型 → 代码生成 → 代码审查 → 单测执行 → 结果回流修复。

五、论述题（本题共 2 小题，每小题 12 分，共 24 分）

19. 论述你对“2027 年大模型 Agent 技术趋势”的理解，并说明你认为最有潜力的 3 个方向及其落地挑战。
趋势：Agent 从单轮问答走向可执行工作流、多 Agent 协作和强工具化。看好 3 个方向：①高可靠 Agentic RAG；
② Multi-Agent 协同与自动软件工程；③垂直行业 Agent 平台化。挑战：评测标准、稳定性/安全性、成本与可观测性。

20. 作为架构师，你如何从 0 到 1 搭建一个高可用、高可观测性的 Agent 平台？请从技术选型、架构设计、监控告警、成本优化等方面展开。
从 0 到 1：技术选型上采用主模型 + 小模型路由，工作流编排 + RAG + 工具网关；架构上分网关层、编排层、
记忆/RAG 层、工具层、审计层；监控上做全链路 trace、成功率/延迟/成本监控与告警；成本上通过缓存、
模型分级、短上下文、批处理优化。

第 1 页（共 8 页）

The “GenAI” Era

Text-to-video generation

Sora

The "GenAI" Era

Text-to-video generation

The “GenAI” Era

Text-to-video generation

Veo3.1

The “GenAI” Era

Text-to-video generation

Seedance 2.0 Pro

The “GenAI” Era

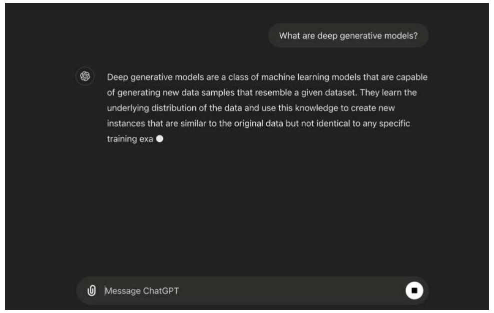
Video-to-video generation

Seedance 2.0 Pro

What are Generative Models?

What do these scenarios have in common?

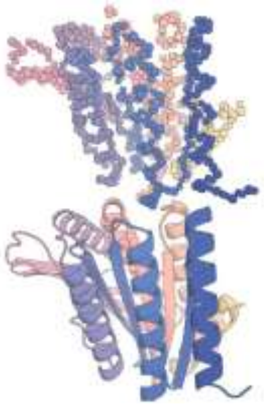
- There are multiple or infinite predictions to one input.
- Some predictions are more “plausible” than some others.
- Training data may contain no exact solution.
- Predictions may be more complex, more informative, and higher-dimensional than input.



Chatbot

Image generation

Video generation



Protein generation

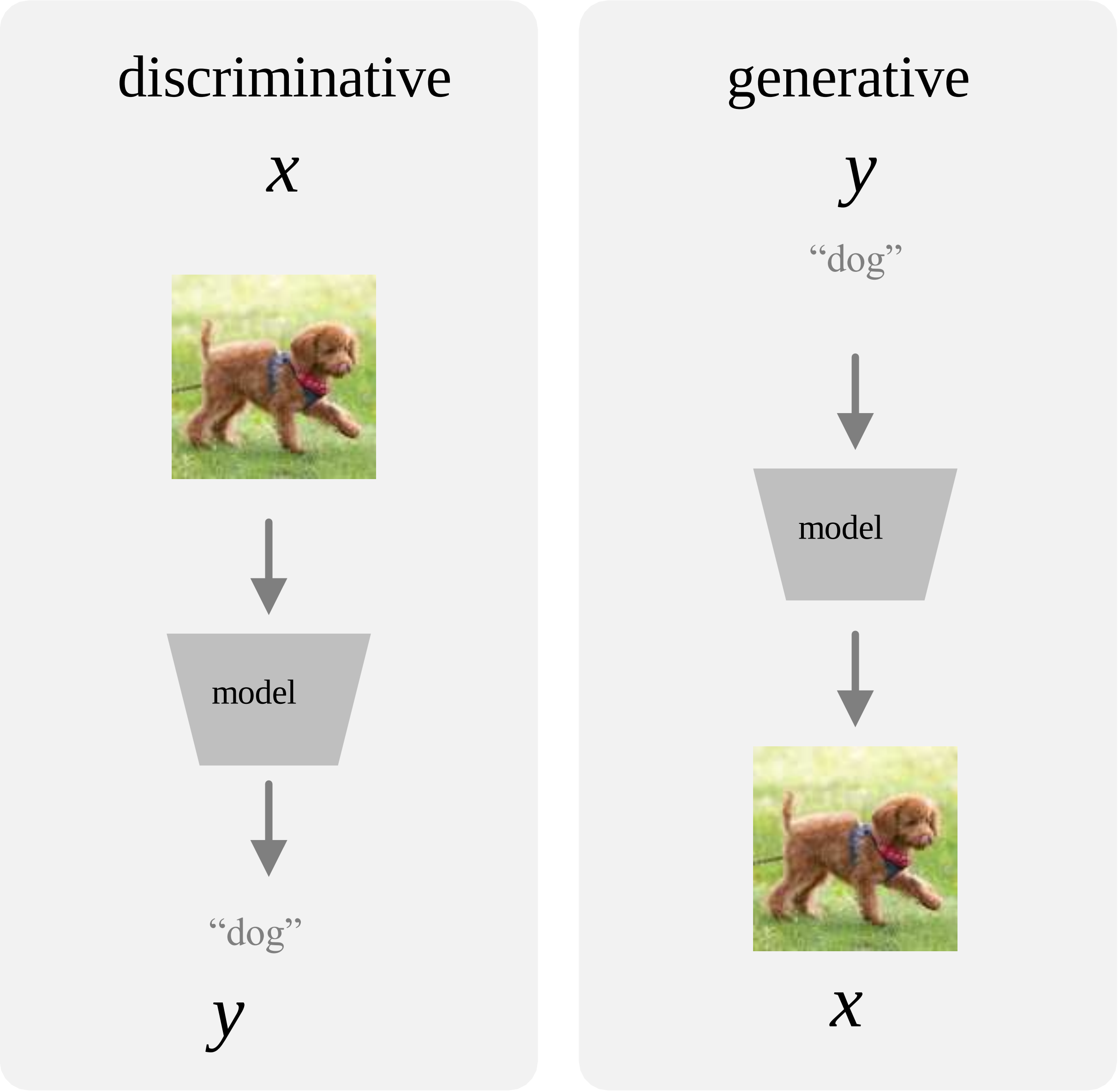
Discriminative vs. Generative models

discriminative

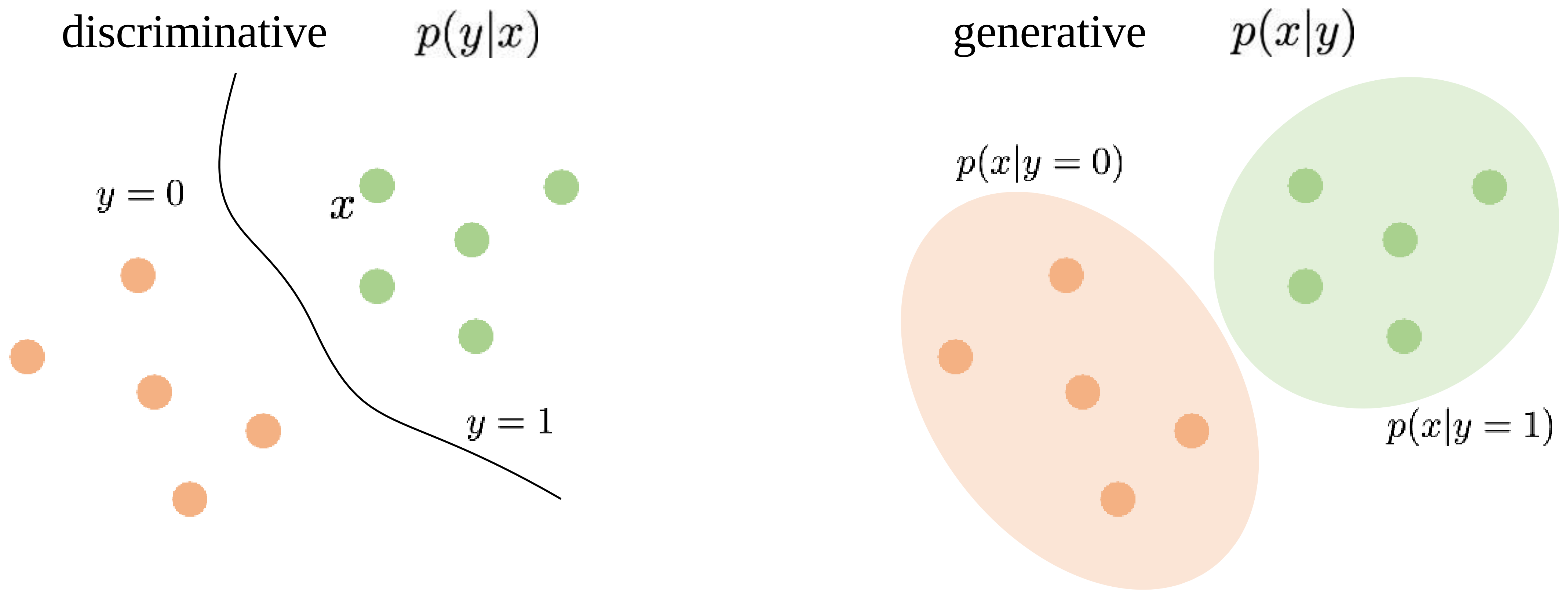
- “sample” x ⇨ “label” y
- one desired output

generative

- “label” y ⇨ “sample” x
- many possible outputs



Discriminative vs. Generative models



- Generative models can be discriminative: Bayes' rule
- Can discriminative models be generative?

- Generative models can be discriminative: Bayes' rule

assuming known prior

$$p(y|x) = p(x|y)\frac{p(y)}{p(x)}$$

discriminative

generative

constant for given x

- Generative models can be discriminative: Bayes' rule

$$\underbrace{p(y|x)}_{\text{discriminative}} = \underbrace{p(x|y)}_{\text{generative}} \frac{p(y)}{p(x)}$$

assuming known prior

constant for given *x*

- Can discriminative models be generative?

$$\underbrace{p(x|y)}_{\text{generative}} = \underbrace{p(y|x)}_{\text{discriminative}} \frac{p(x)}{p(y)}$$

still need to model prior distribution of *x*

constant for given *y*

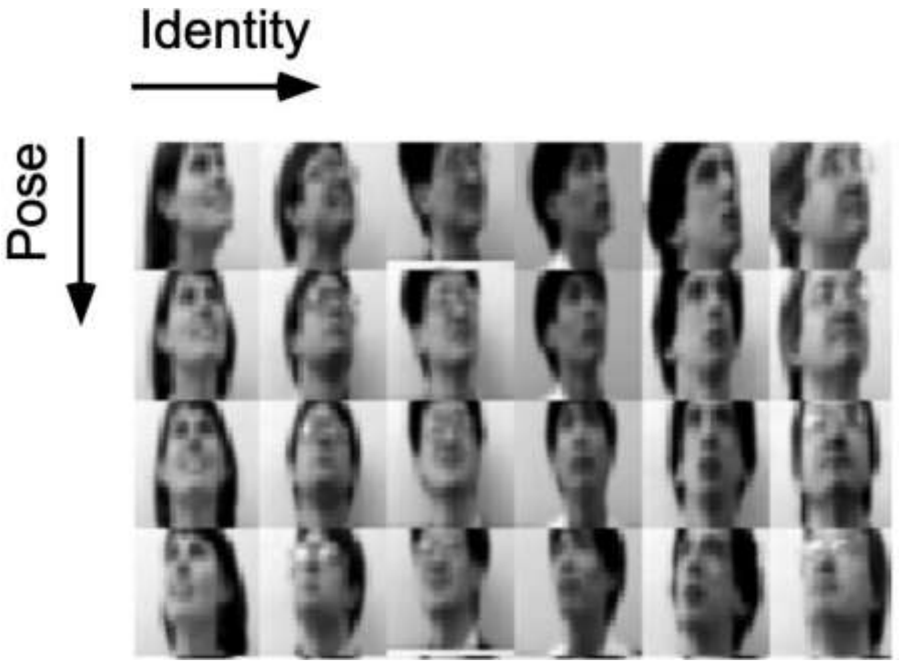
- The challenge is about representing and predicting distributions

Probabilistic modeling

- Where does probability come from?
- Assuming underlying distributions of data generation process

example:

- latent factors z (pose, lighting, scale, ...)
- z has simple distributions
- observations x are rendered by a "world model" that's a function on z
- observations x have complex distributions



- Probability is part of the modeling.

Figure from: W. T. Freeman, J. B. Tenenbaum, "Learning Bilinear Models for Two-Factor Problems in Vision", 1996

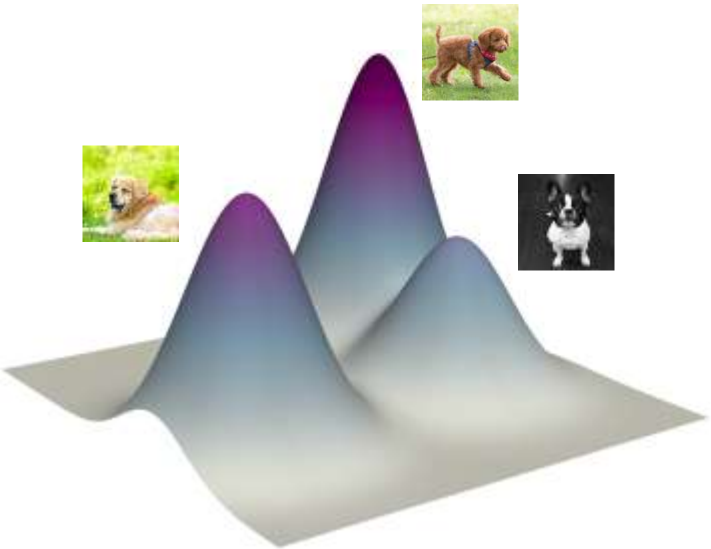
Generative models w/ probabilistic modeling

data

Generative models w/ probabilistic modeling

data

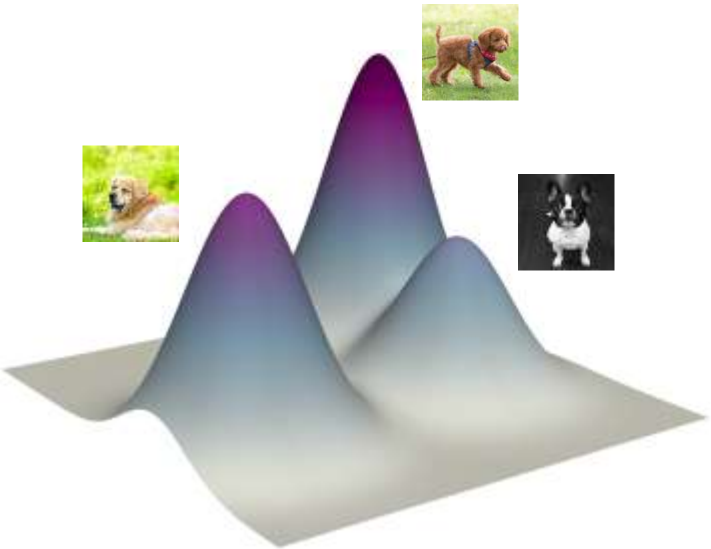
distribution of data



- This is already part of the modeling

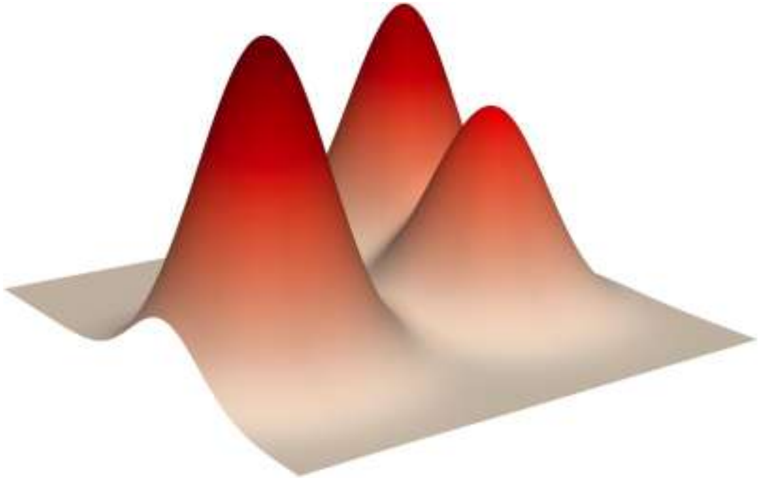
Generative models w/ probabilistic modeling

data



distribution of data

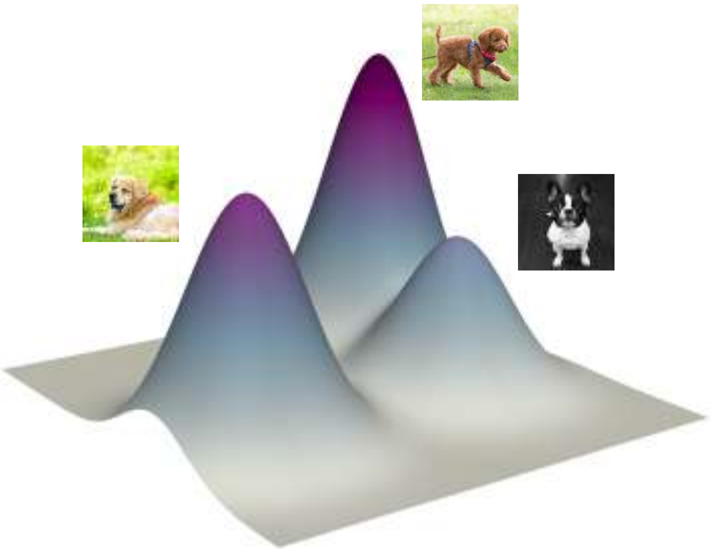
estimated distribution of data



- Optimize a loss function

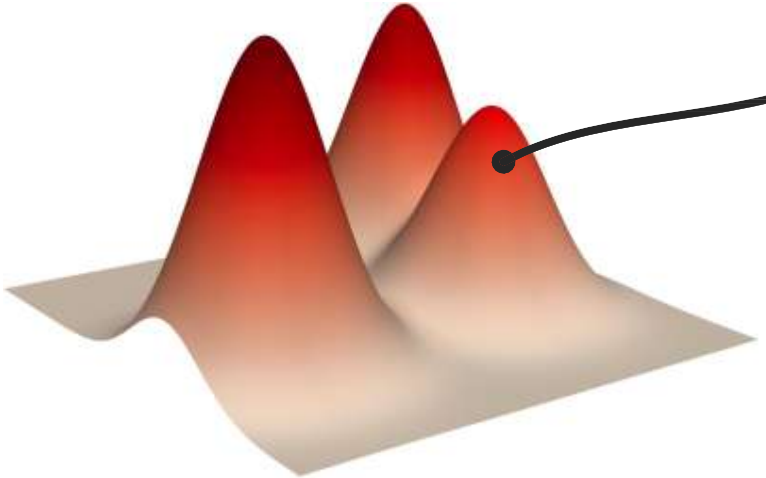
Generative models w/ probabilistic modeling

data



distribution of data

estimated distribution of data



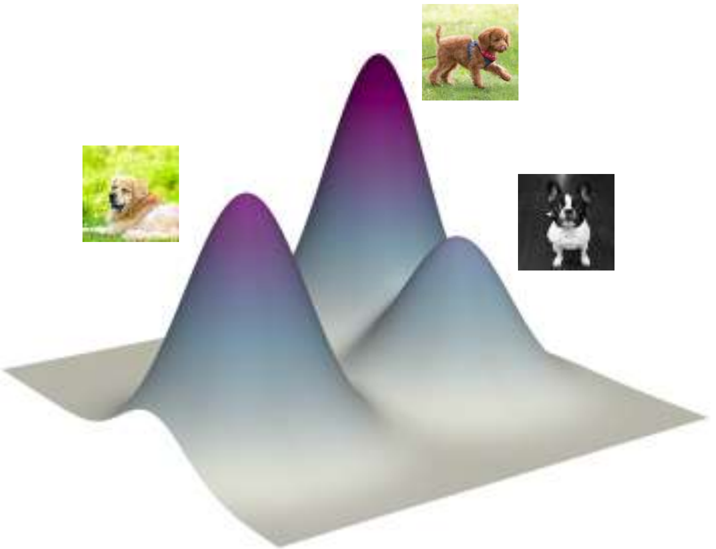
sample new “data”

$$x' \sim p(x|y)$$

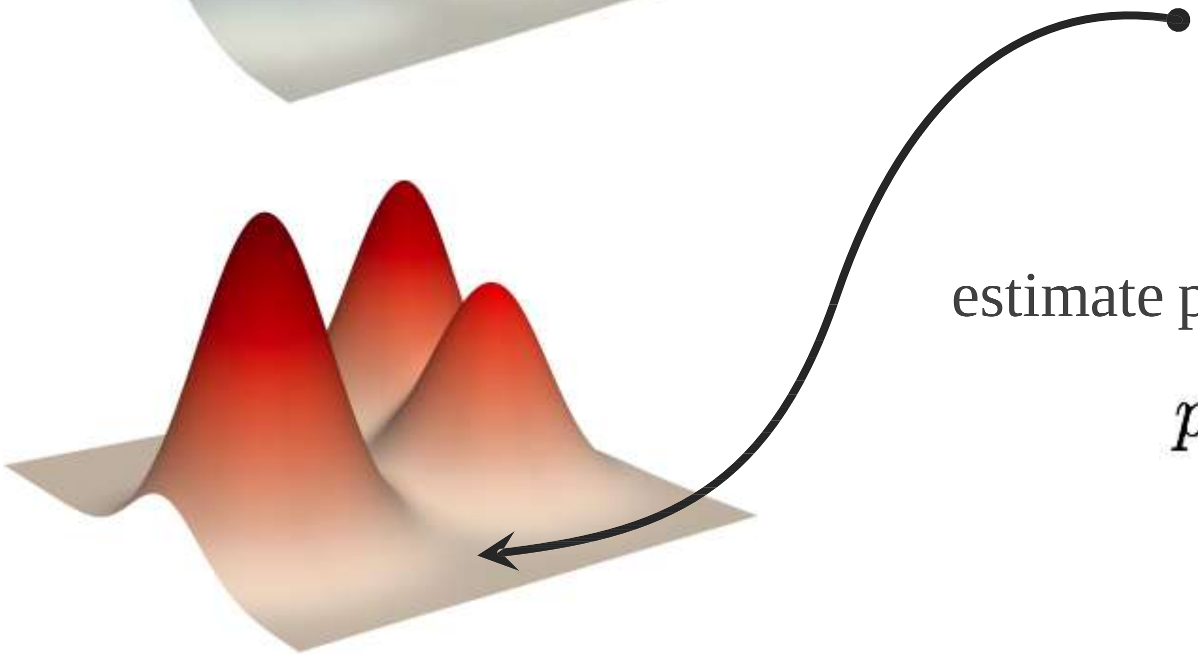
Generative models w/ probabilistic modeling

data

distribution of data



estimated distribution of data



estimate prob density

$p(x|y)$ = ?

Generative models w/ probabilistic modeling

Notes:

- Generative models involve statistical models which are often designed and derived by humans.
- Probabilistic modeling is not just the work of neural nets.
- Probabilistic modeling is a popular way, but not the only way.
- "*All models are wrong, but some are useful*." - George Box

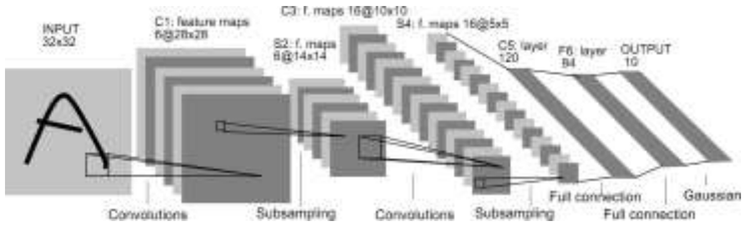
<u>Deep</u> Generative Models

- Deep learning is representation learning
- Learning to represent data instances
  - map data to feature:
  - minimize loss w/ targe

$$x \rightarrow f(x)$$

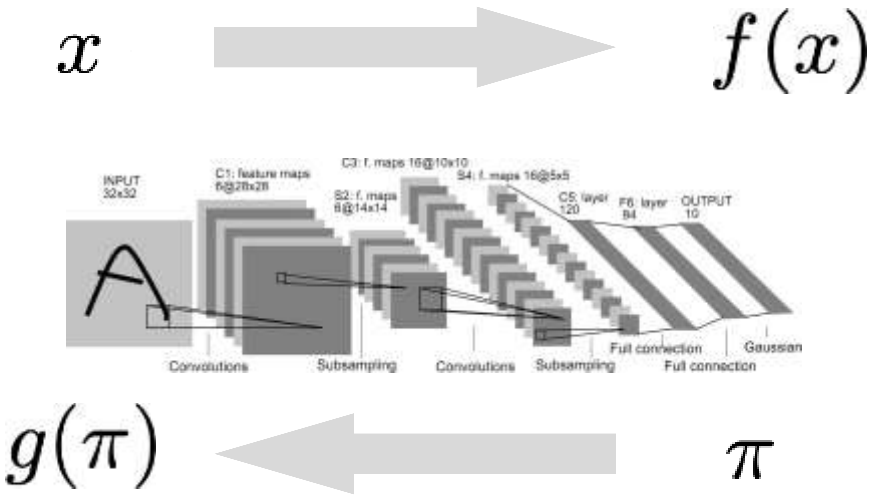
$$\mathcal{L}(y, f(x))$$

$$x \longrightarrow f(x)$$



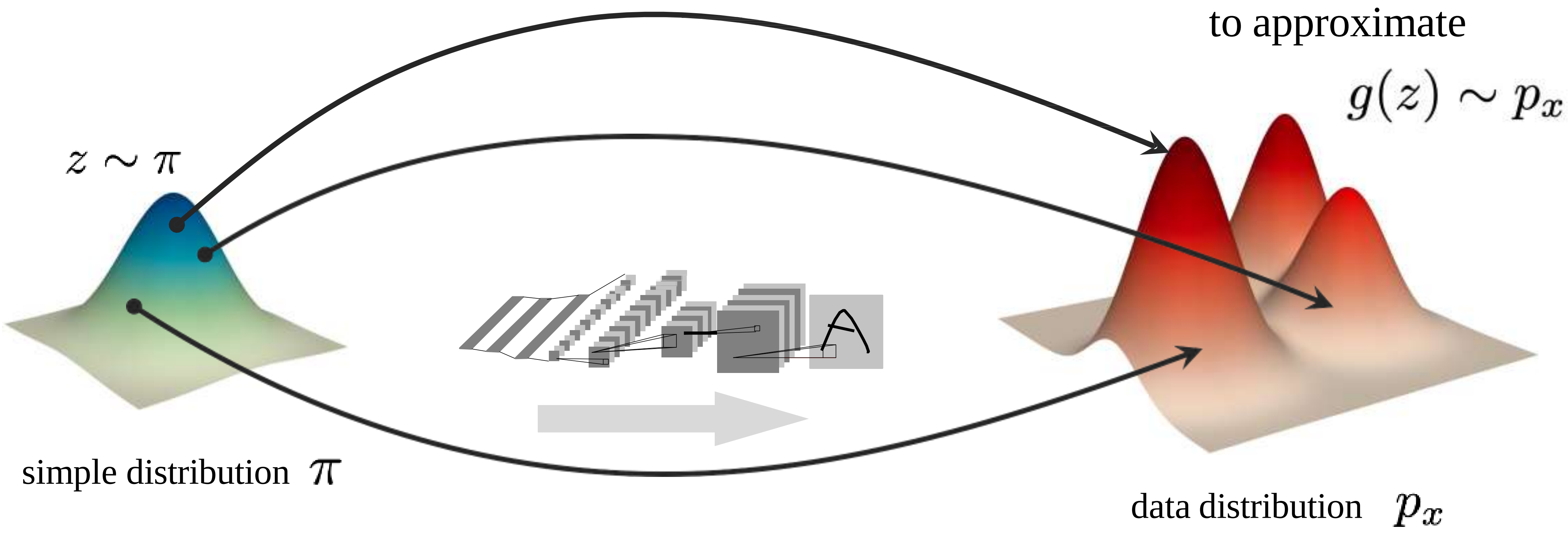
Deep Generative Models

- Deep learning is representation learning
- Learning to represent data instances
  - map data to feature: $x \rightarrow f(x)$
  - minimize loss w/ target: $\mathcal{L}(y, f(x))$
- Learning to represent probability distributions
  - map a simple distribution (Gaussian/uniform) to a complex one: $\pi \rightarrow g(\pi)$
  - minimize loss w/ data distribution: $\mathcal{L}(p_x, g(\pi))$
- Often perform both together



Learning to represent probability distributions

- From simple to complex distributions

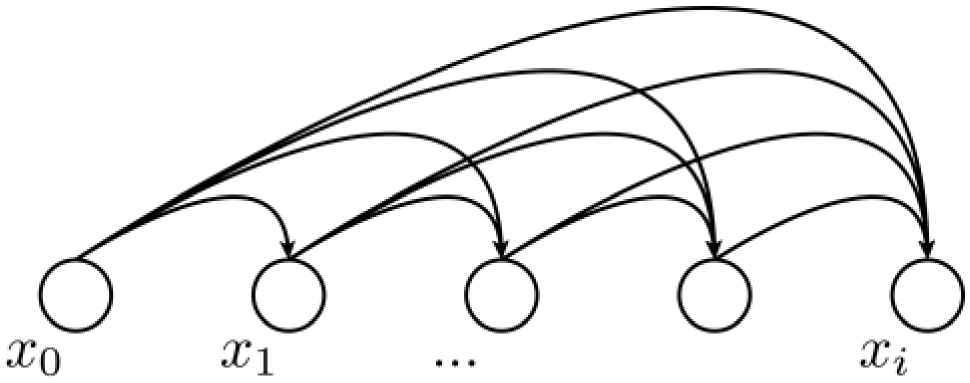


Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

Case study:
Autoregressive model

This dependency graph is designed (not learned).

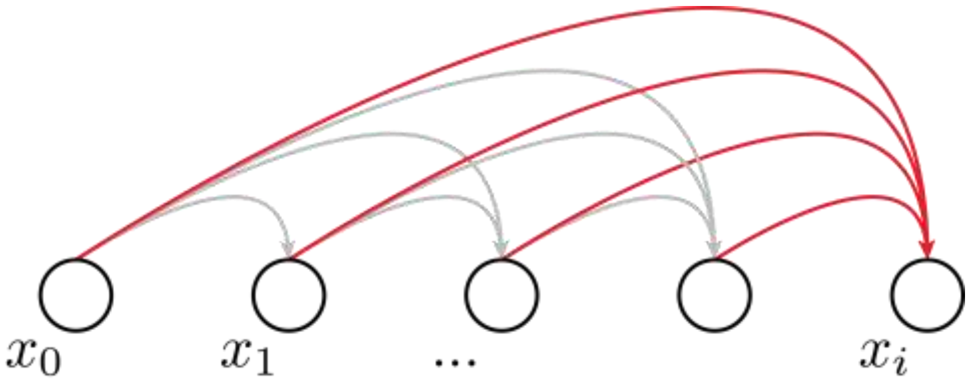


Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

Case study:
Autoregressive model

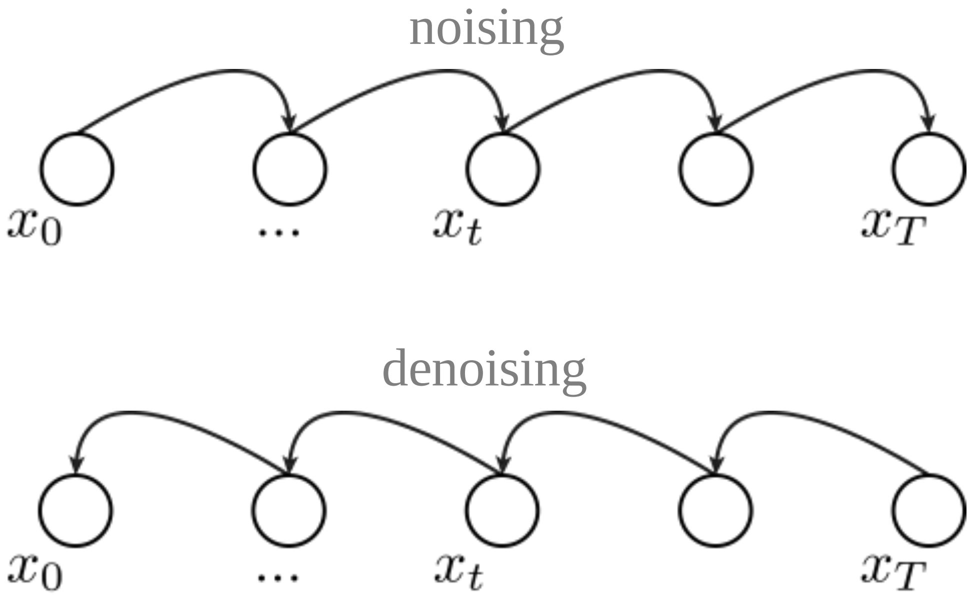
The mapping function is learned
(e.g., Transformer)



Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

Case study:
Diffusion model

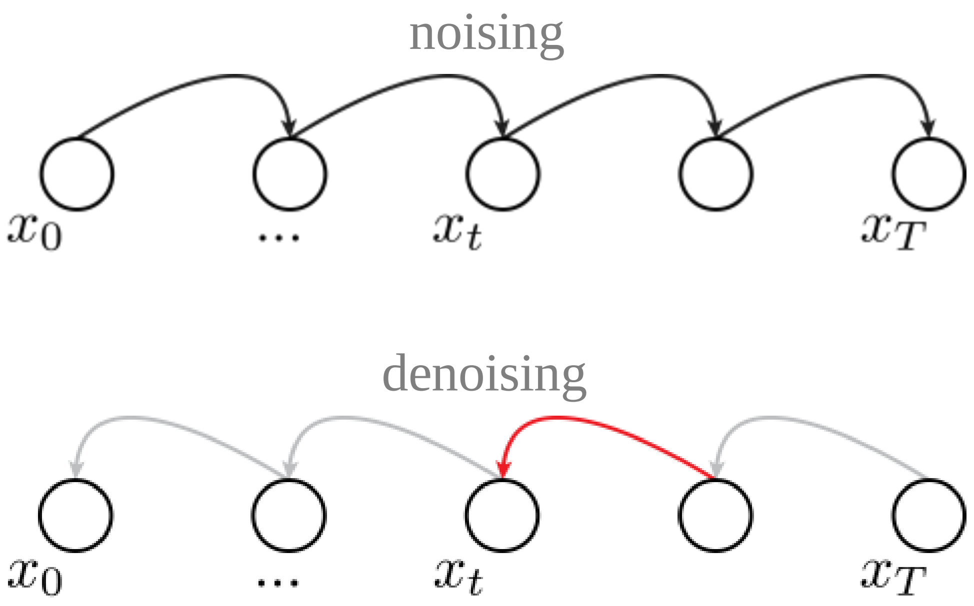


This dependency graph is designed (not learned).

Learning to represent probability distributions

- Not all parts of distribution modeling is done by learning

Case study:
Diffusion model



The mapping function is learned
(e.g., Unet)

Deep Generative Models may involve:

- Formulation:
  - formulate a problem as probabilistic modeling
  - decompose complex distributions into simple and tractable ones
- Representation: deep neural networks to represent data and their distributions
- Objective function: to measure how good the predicted distribution is
- Optimization: optimize the networks and/or the decomposition
- Inference:
  - sampler: to produce new samples
  - probability density estimator (optional)

Formulating Real-world Problems as Generative Models

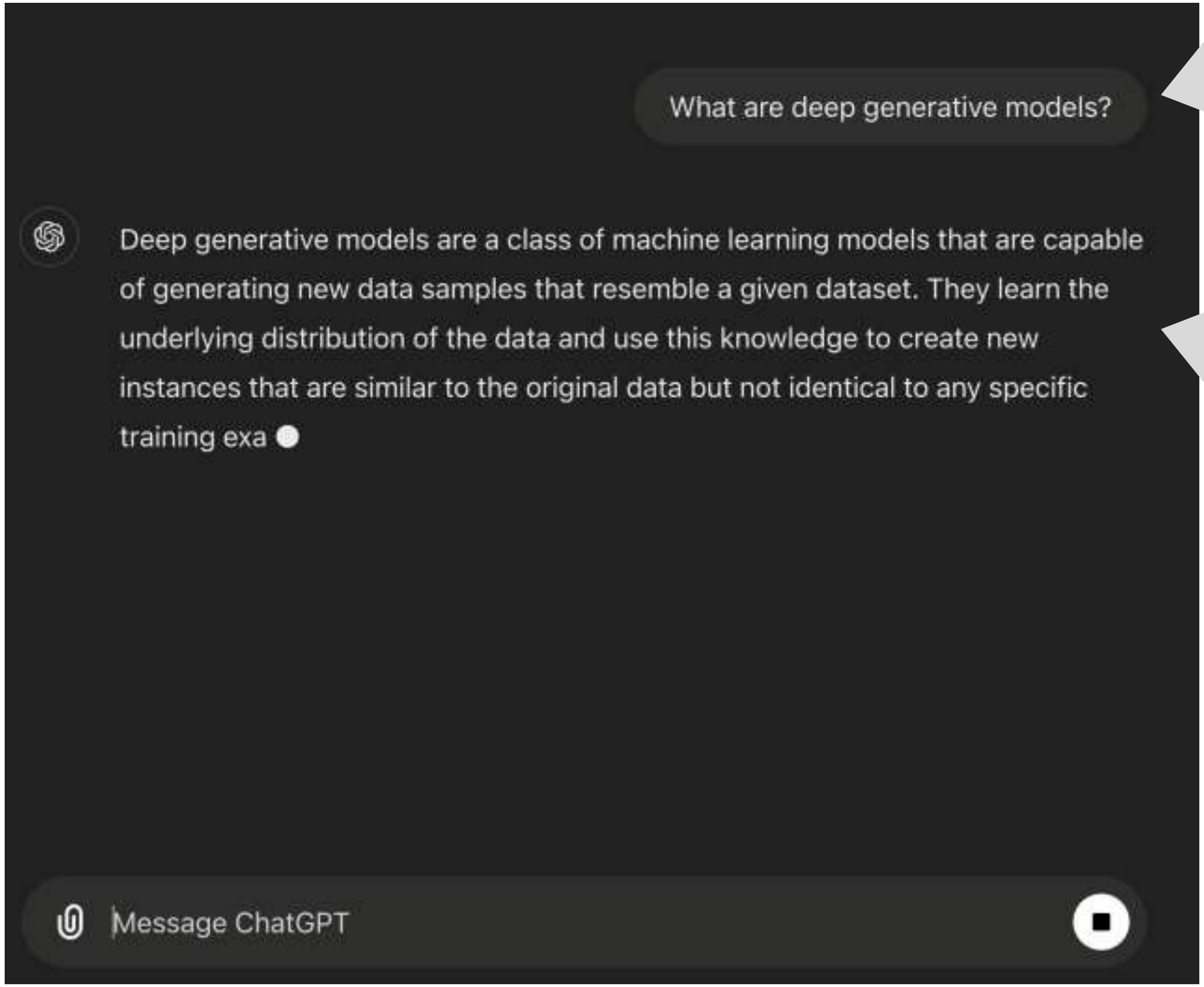
Formulating Real-world Problems as Generative Models

- Generative models are about $p(x|y)$

What can be *y*?	What can be *x*?
• condition	• “data”
• constraint	• samples
• labels	• observations
• attributes	• measurements
• more abstract	• more concrete
• less informative	• more informative

Case study: Formulating as $p(x|y)$

- Natural language conversation



y: prompt

x: response of the chatbot

Case study: Formulating as $p(x|y)$

- Text-to-image/video generation

Prompt: ~*~aesthetic~*~ #boho #fashion, full-body 30-something woman laying on microfloral grass, candid pose, overlay reads Stable Diffusion 3.5, cheerful cursive typography font.

y: text prompt

x: generated visual content

Image generated by Stable Diffusion 3.5

Case study: Formulating as $p(x|y)$

- Text-to-3D structure generation

x: generated 3D structures

y: text prompt

Figure credit: Tang, et al. LGM: Large Multi-View Gaussian Model for High-Resolution 3D Content Creation. ECCV 2024

Case study: Formulating as $p(x|y)$

- Image-to-3D structure generation

x: generated 3D structures

y: image prompt

https://stability.ai/news/stable-point-aware-3d

Case study: Formulating as $p(x|y)$

- Image-to-3D structure generation

x: generated 3D structures

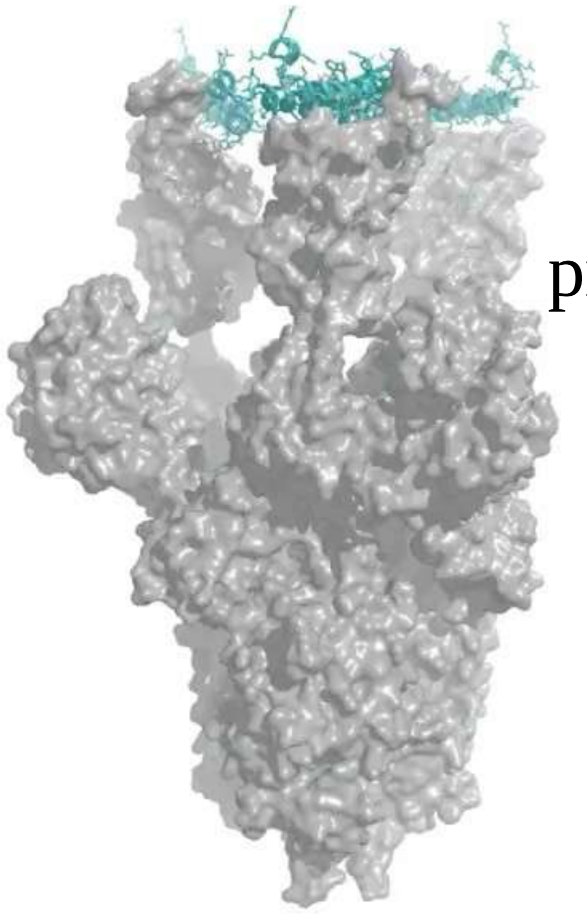
y: image prompt

https://stability.ai/news/stable-point-aware-3d

Case study: Formulating as $p(x|y)$

- Protein structure generation

y: condition/constraint (e.g., symmetry)



x: generated protein structures

Watson, et al. De novo design of protein structure and function with RFdiffusion, Nature 2023

Case study: Formulating as $p(x|y)$

- "Unconditional" image generation

y: an implicit condition

"images following CIFAR10 distribution"

x: generated CIFAR10-like images

- $p(x|y)$: images ~ CIFAR10
- $p(x)$: all images

Images generated by: Karras, et al. Elucidating the Design Space of Diffusion-Based Generative Models, NeurIPS 2022

Case study: Formulating as $p(x|y)$

- Image captioning

y: an image as the "condition"

x: plausible descriptions conditioned on the image

```
a baseball player with a catcher and umpire on top of a baseball field.
a baseball player is sliding into a base.
a baseball player swings at a pitch with the pitcher and umpire behind him.
baseball player with bat in the baseball game.
a batter in the process on the bat in a baseball game.
```

figure credit: https://github.com/GoogleCloudPlatform/asl-ml-immersion/blob/master/notebooks/multi_modal/solutions/image_captioning.ipynb

Case study: Formulating as $p(x|y)$

- Chatbot with visual inputs

User What is unusual about this image?

Source: https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg

GPT-4 The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

y: image and text prompt

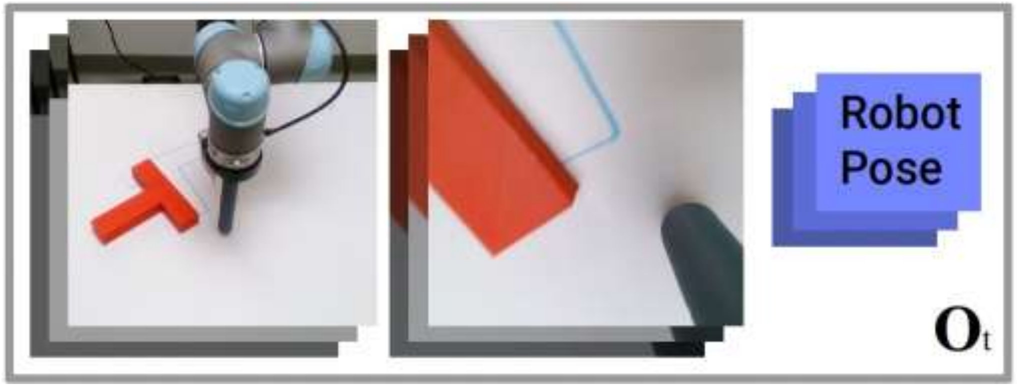
x: response of the chatbot

Figure from: GPT-4 Technical Report, 2023

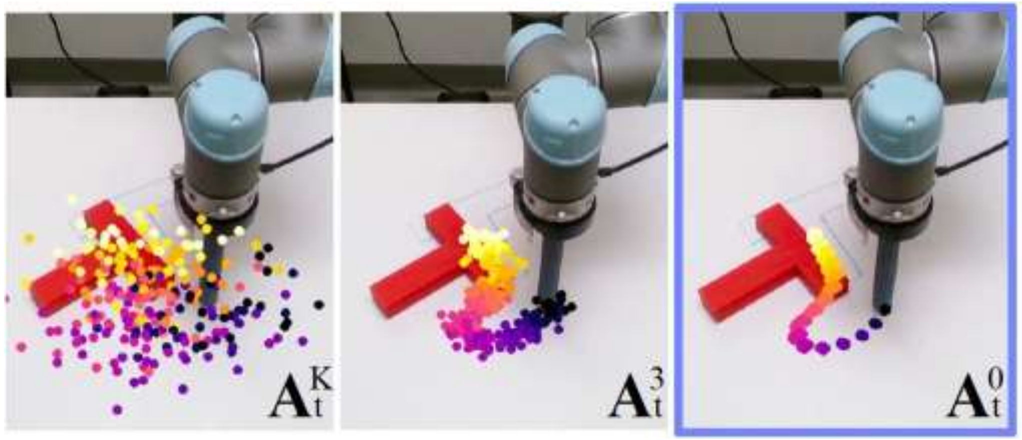
Case study: Formulating as $p(x|y)$

- Policy Learning in Robotics

y: visual and other sensory observations



x: policies
(probability of actions)



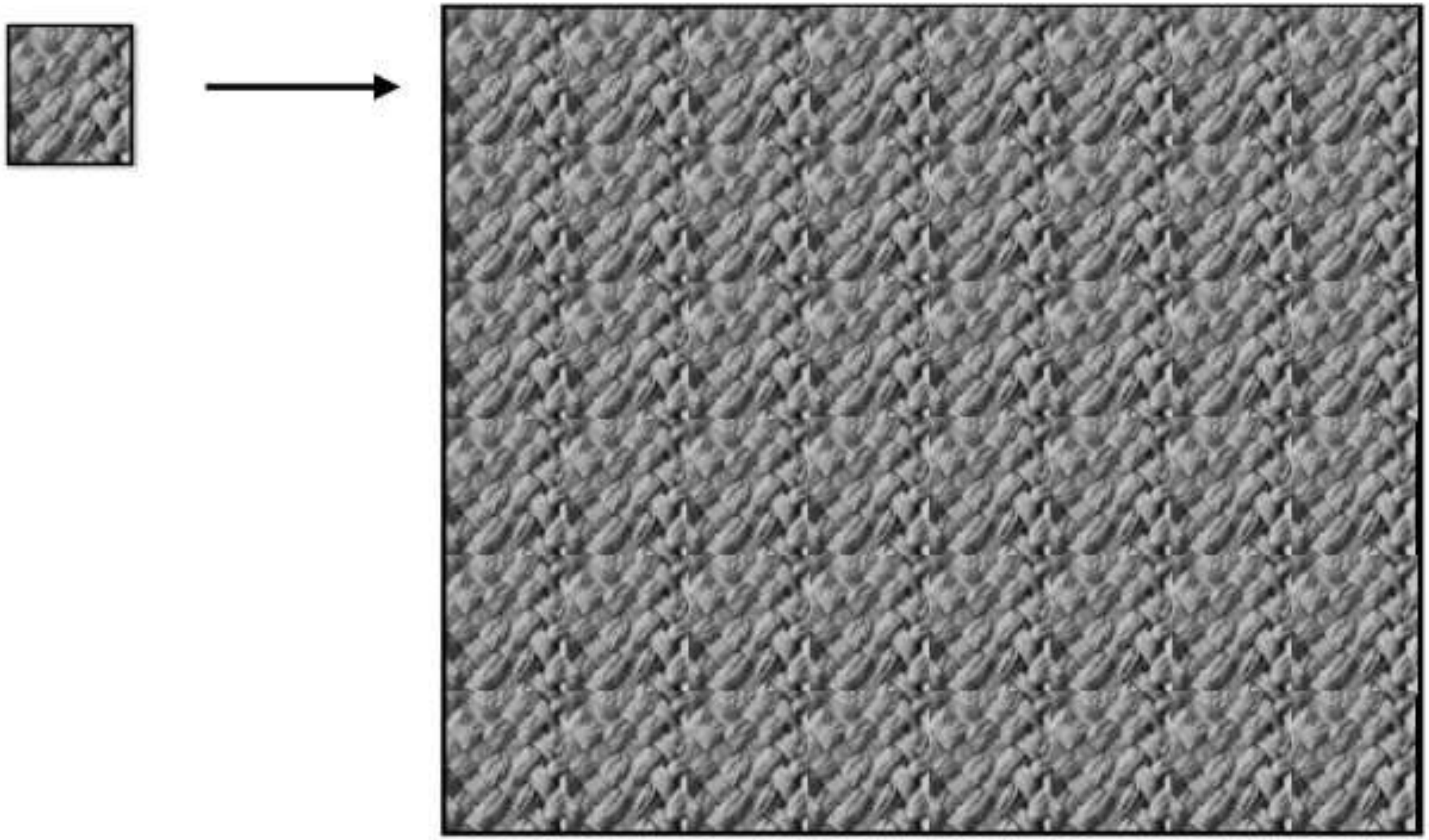
Chi, et al. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, RSS 2023

Formulating Real-world Problems as Generative Models

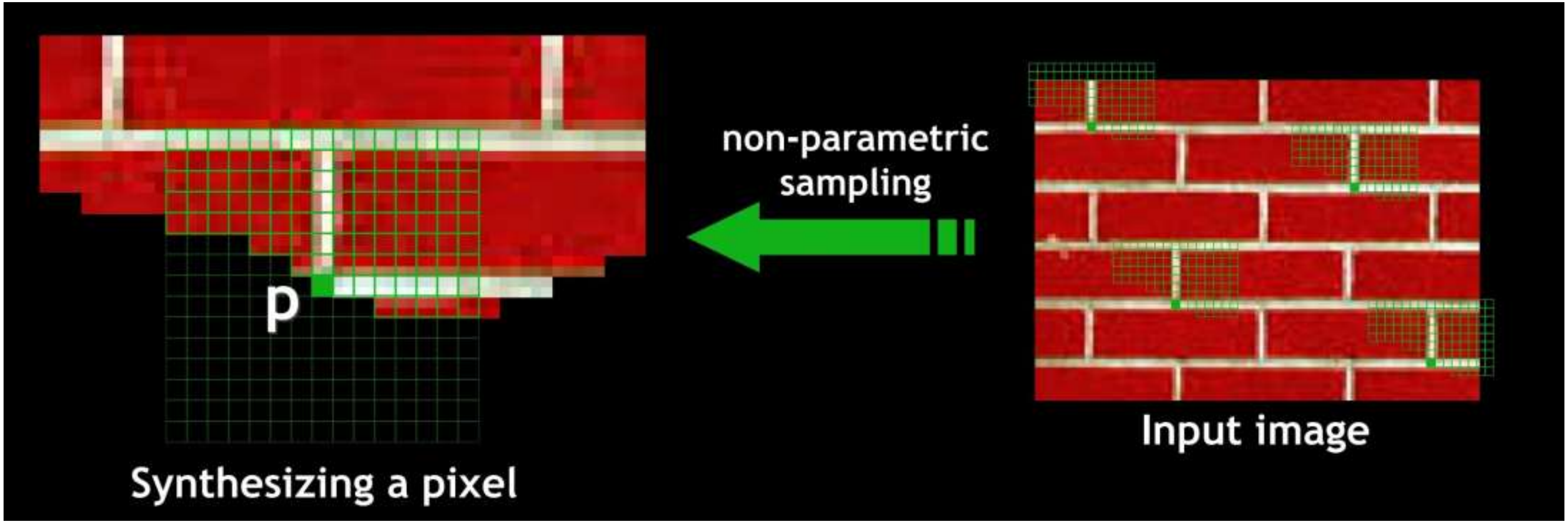
- Generative models are about $p(x|y)$
- Many problems can be formulated as generative models
- What's *x*? What's *y*?
- How to represent *x*, *y*, and their dependence?

Image Manipulation with Example

How to Generate Images?

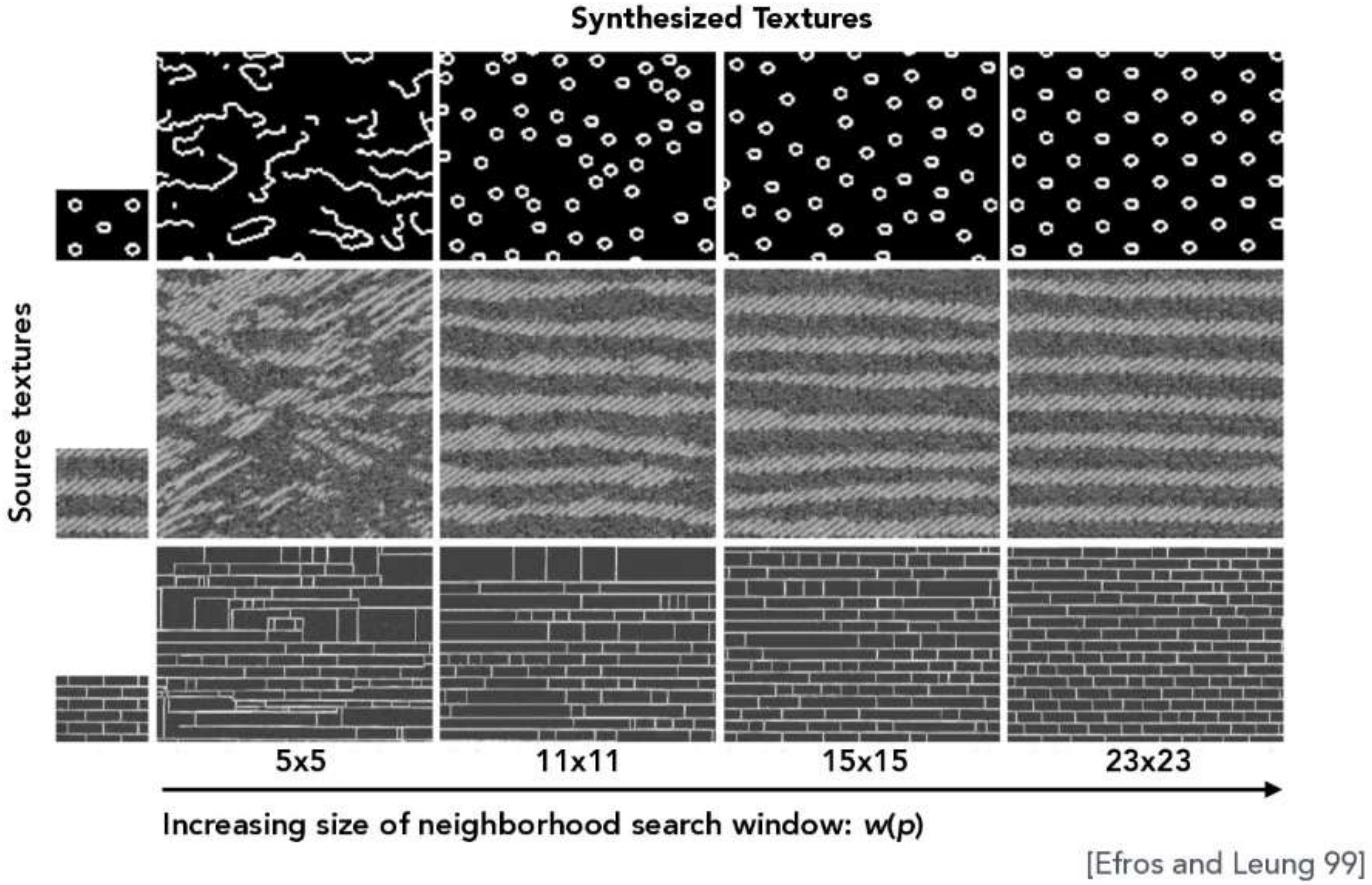


Generative Models before the “GenAI” Era



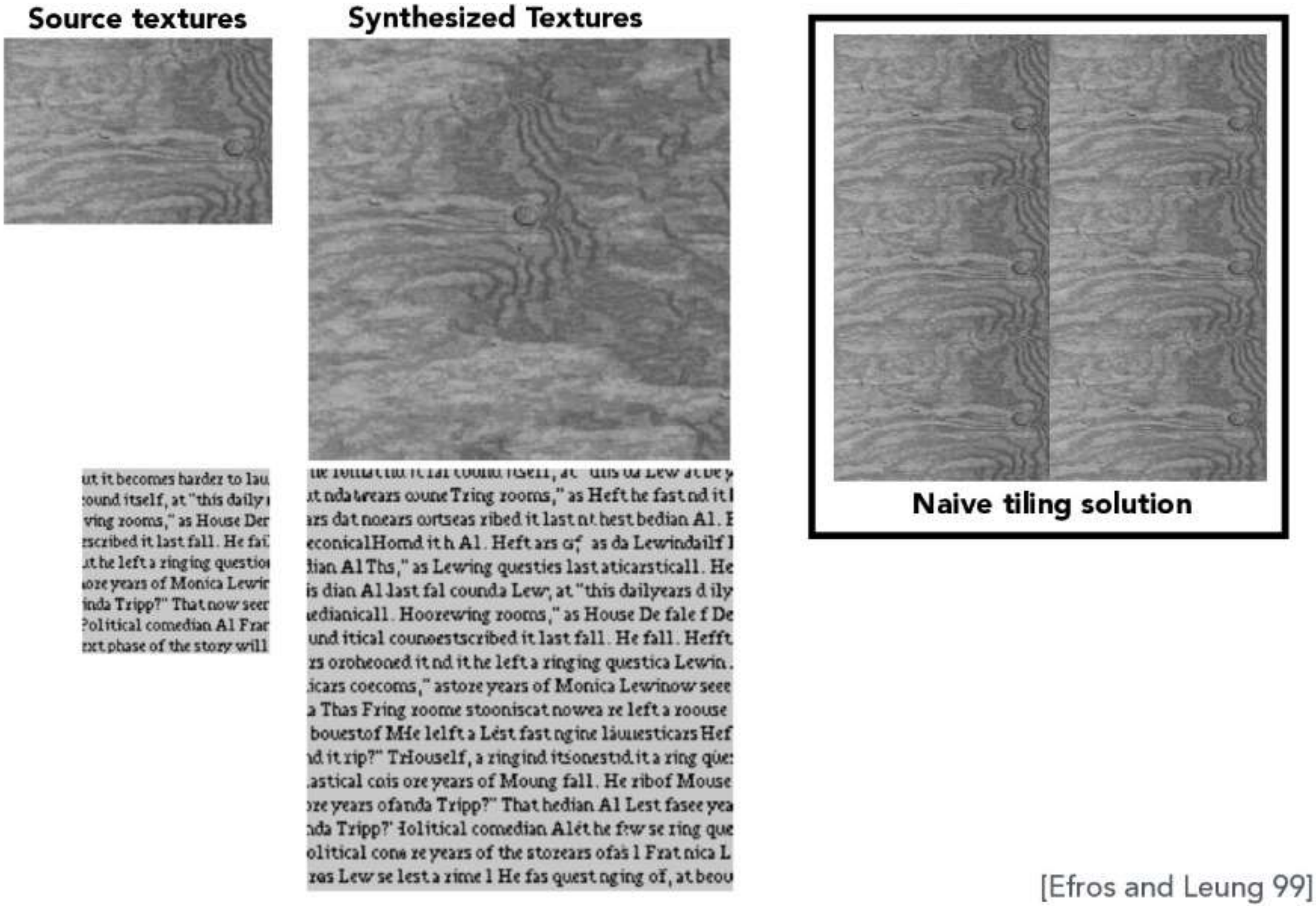
Texture Synthesis by Non-parametric Sampling, ICCV 1999

Generative Models before the “GenAI” Era



Texture Synthesis by Non-parametric Sampling, ICCV 1999

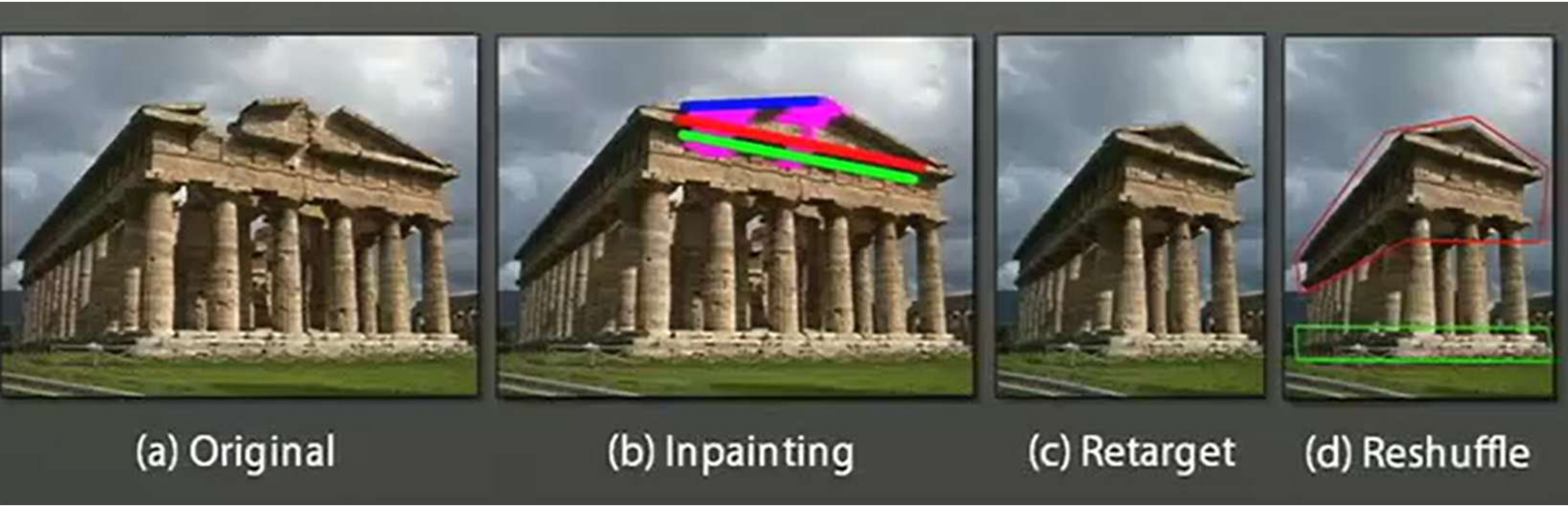
Generative Models before the "GenAI" Era



Texture Synthesis by Non-parametric Sampling, ICCV 1999

PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



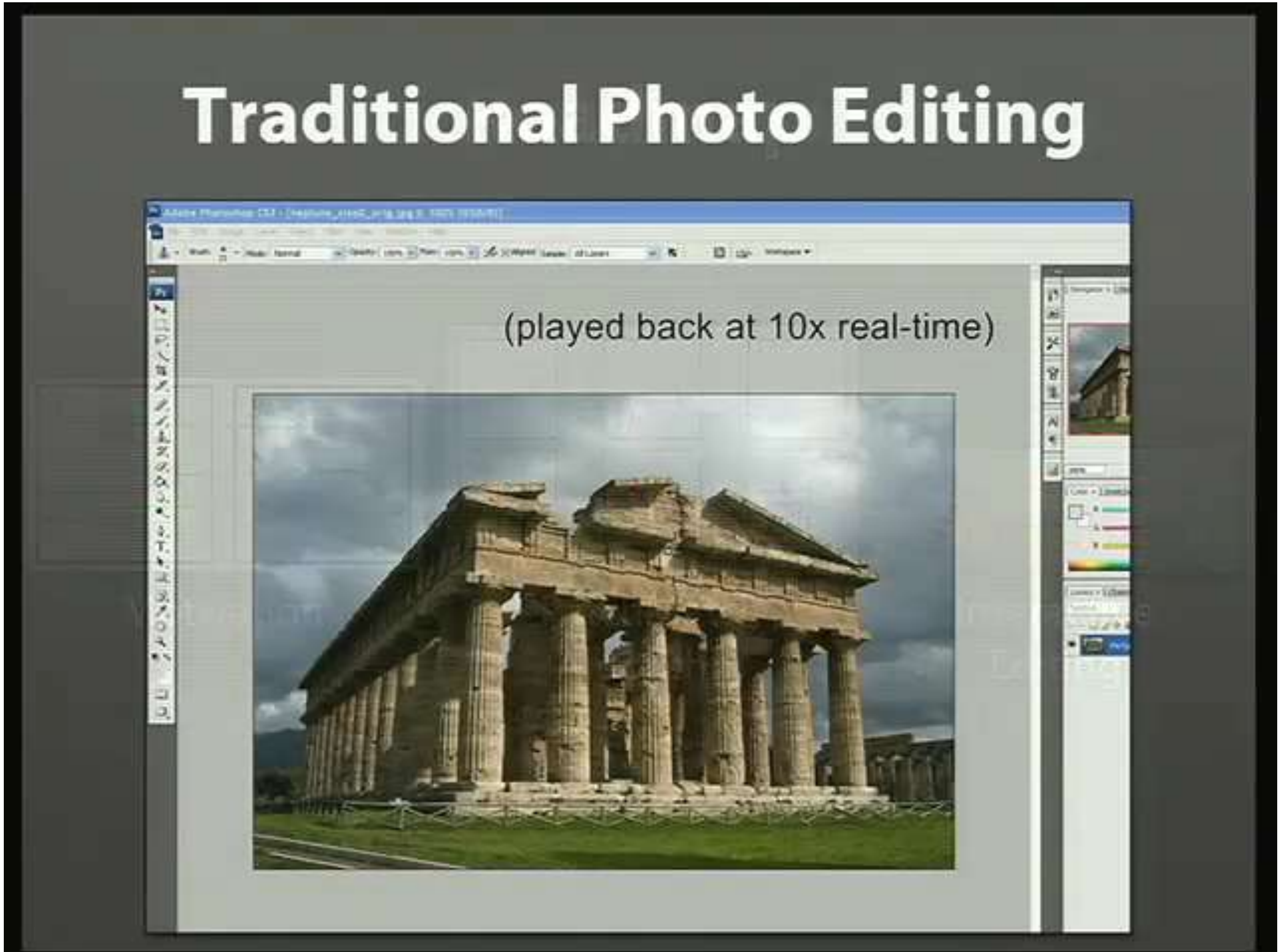
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.

(a) Original (b) Inpainting (c) Retarget (d) Reshuffle

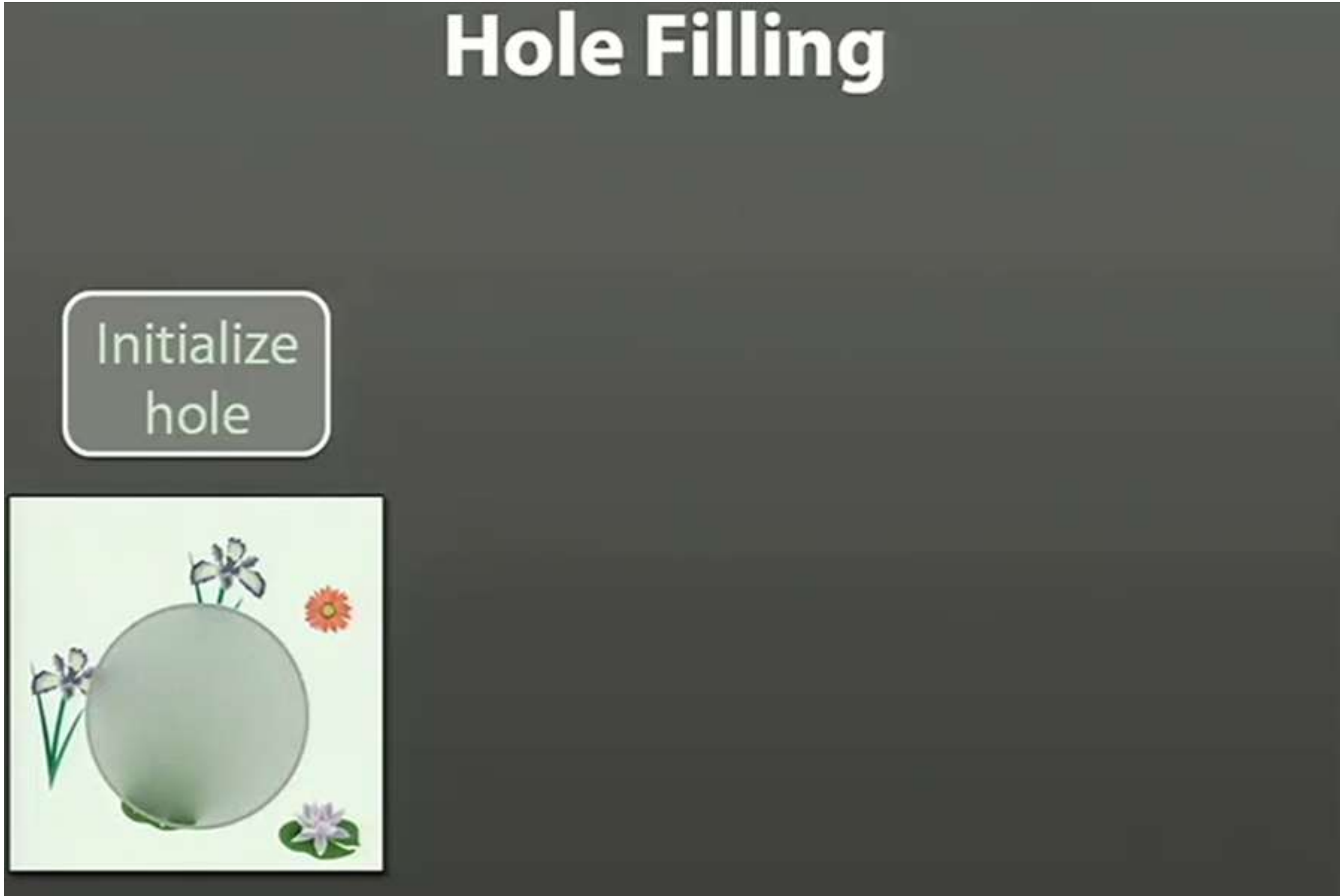
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



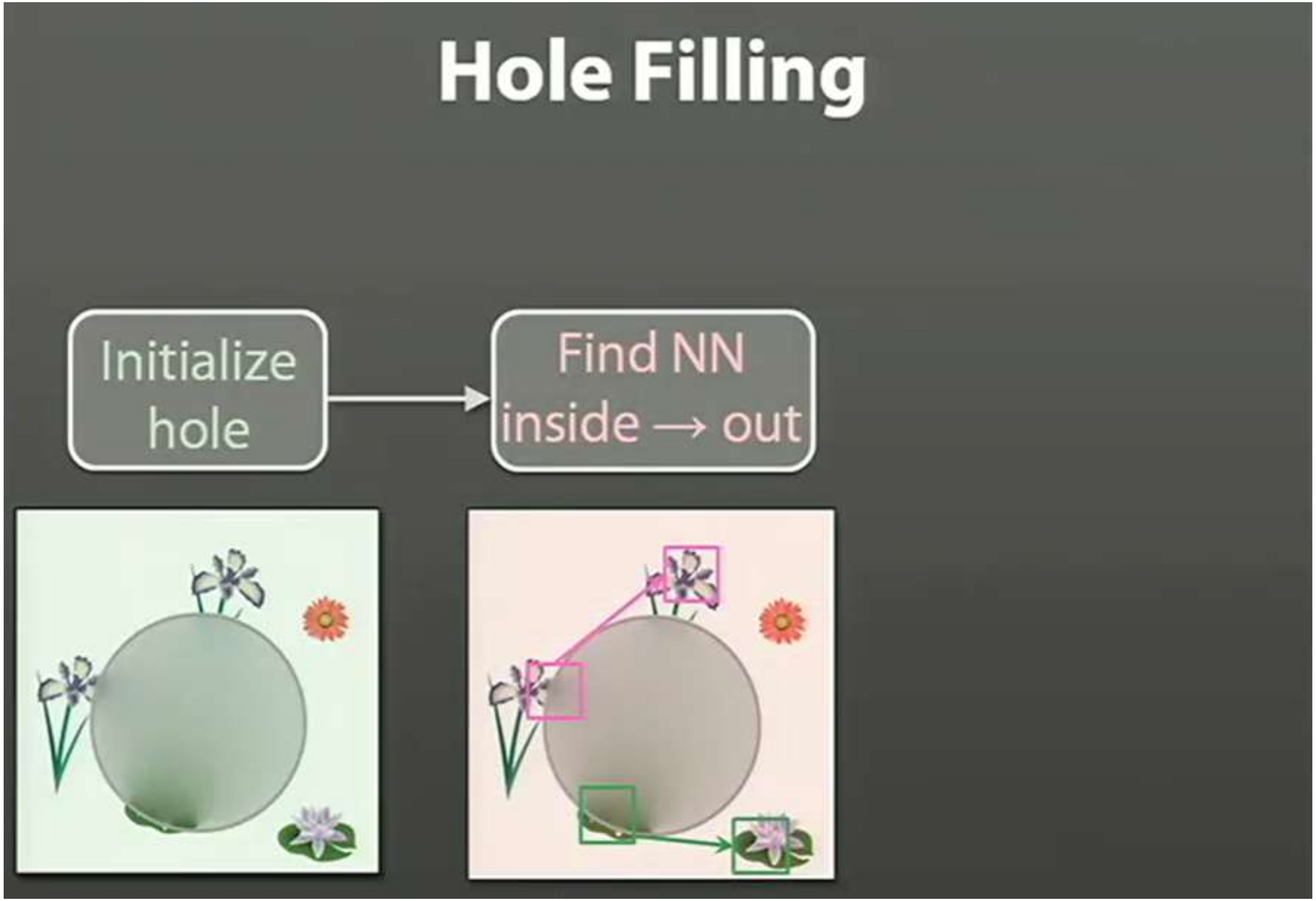
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



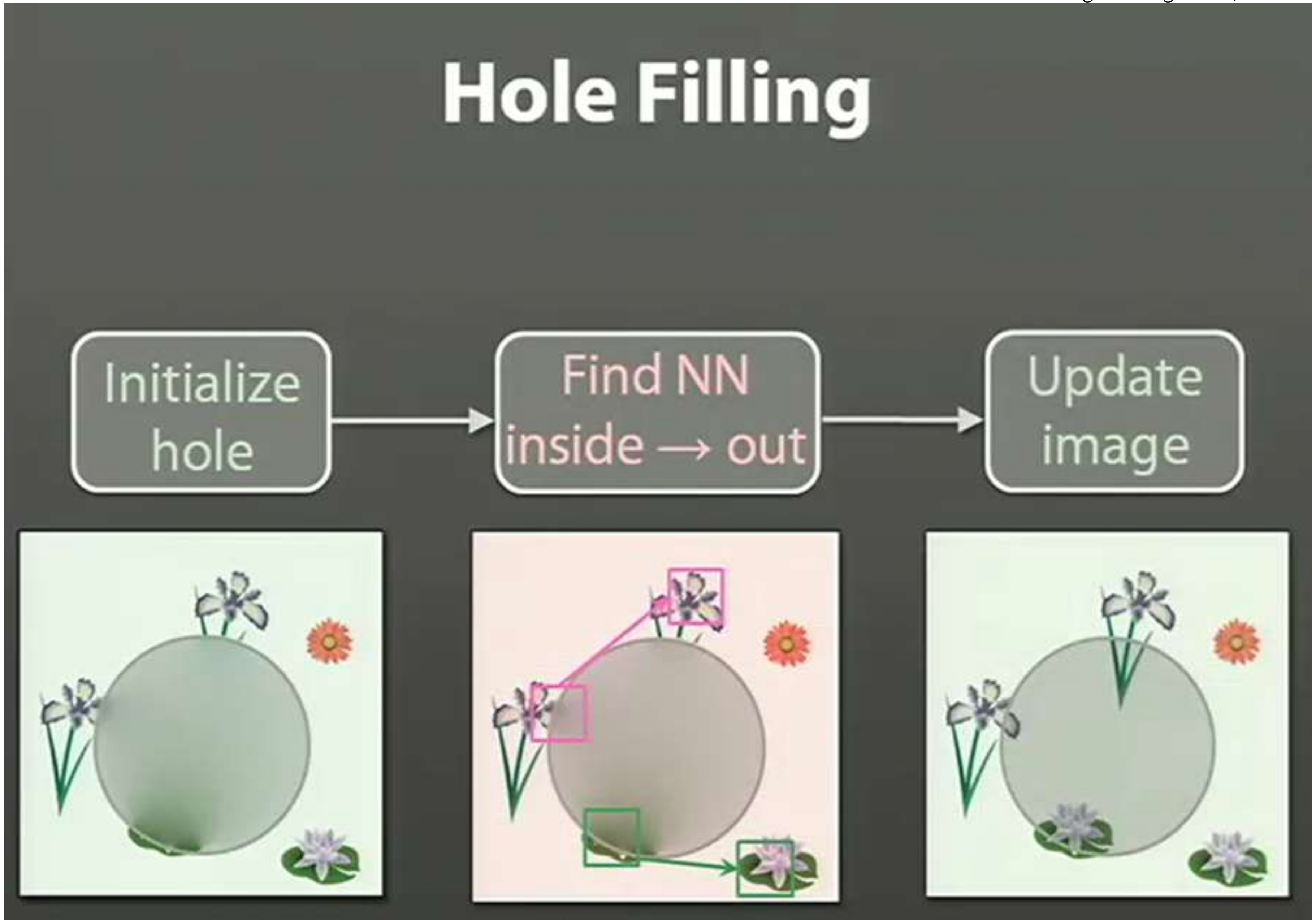
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



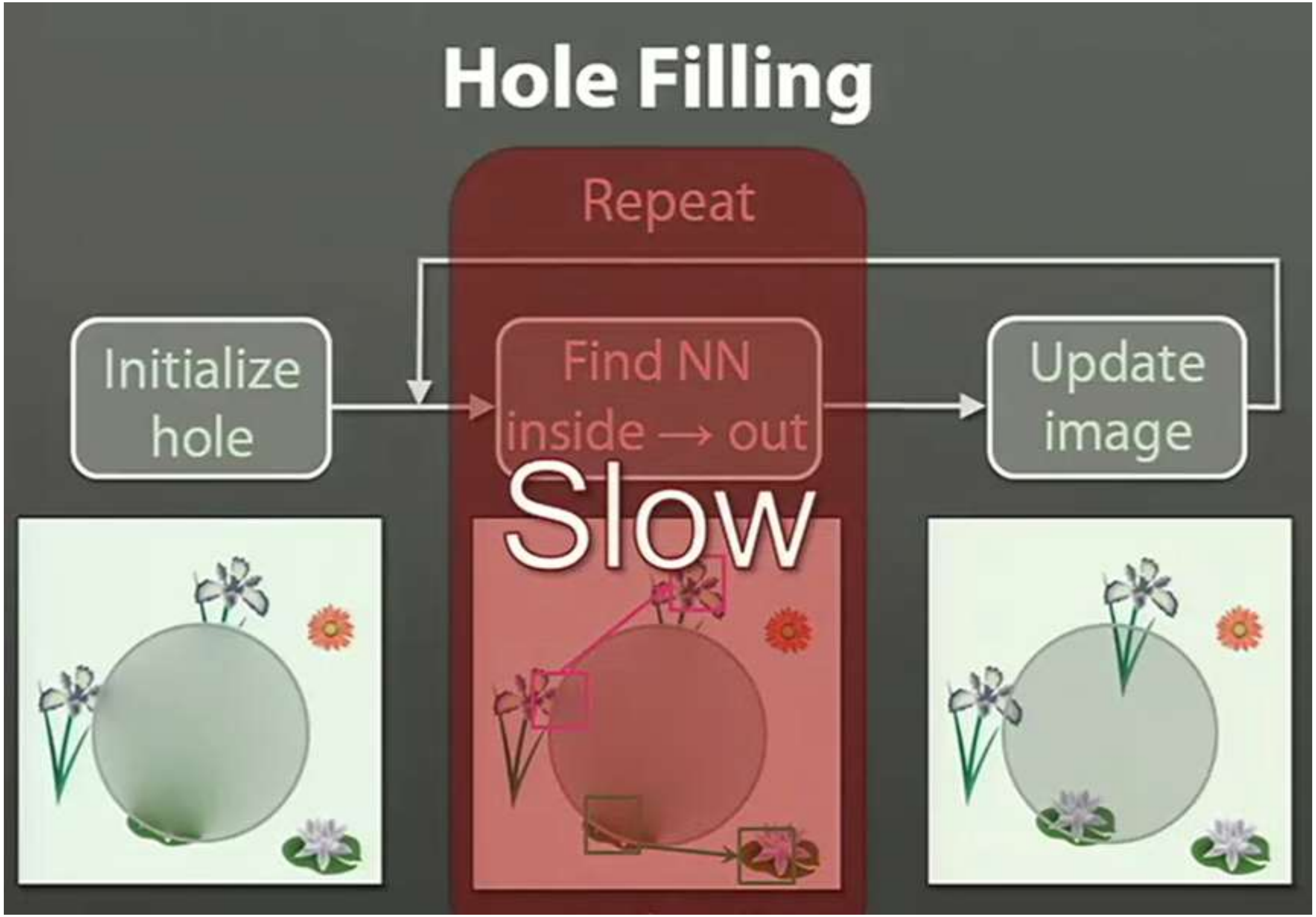
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



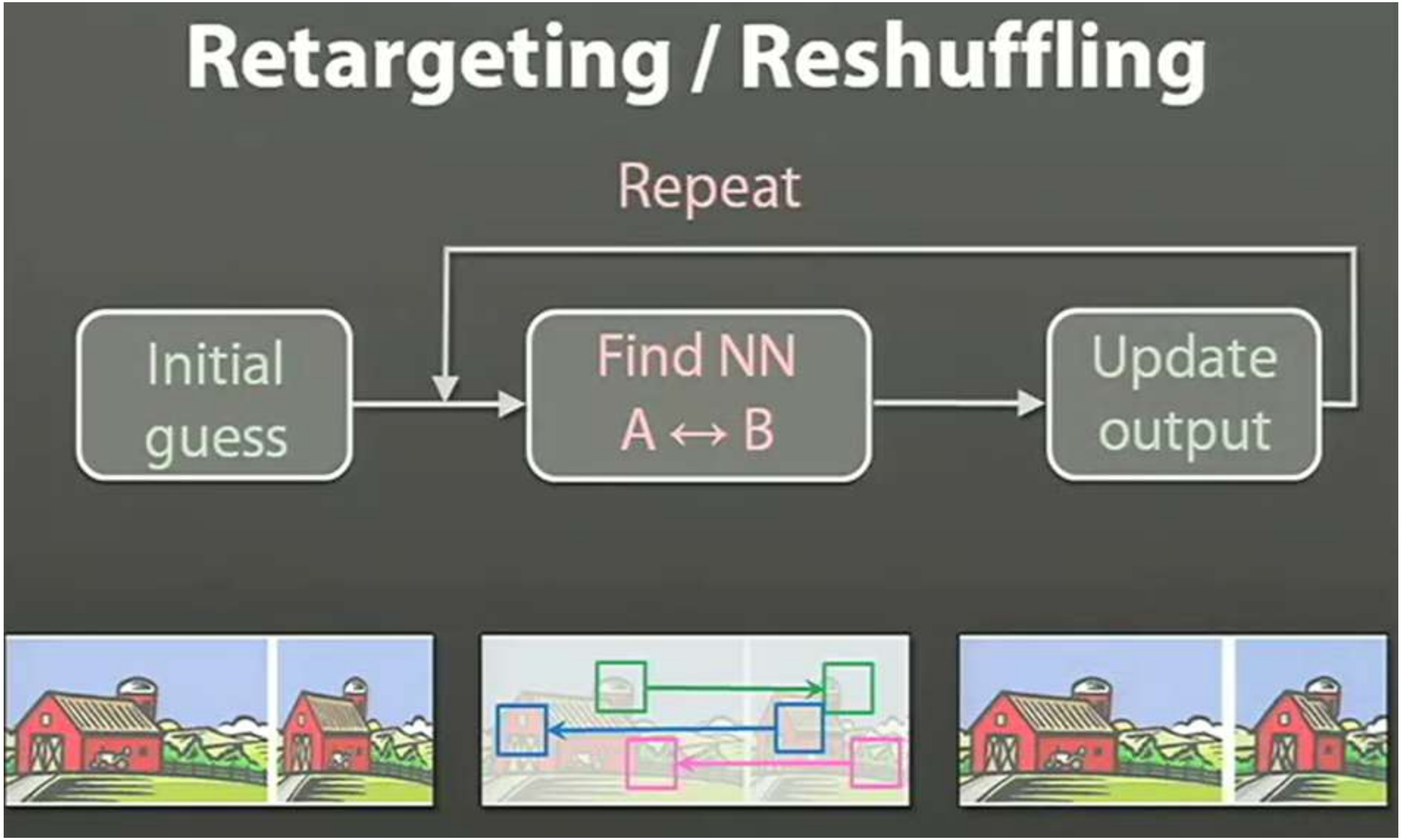
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



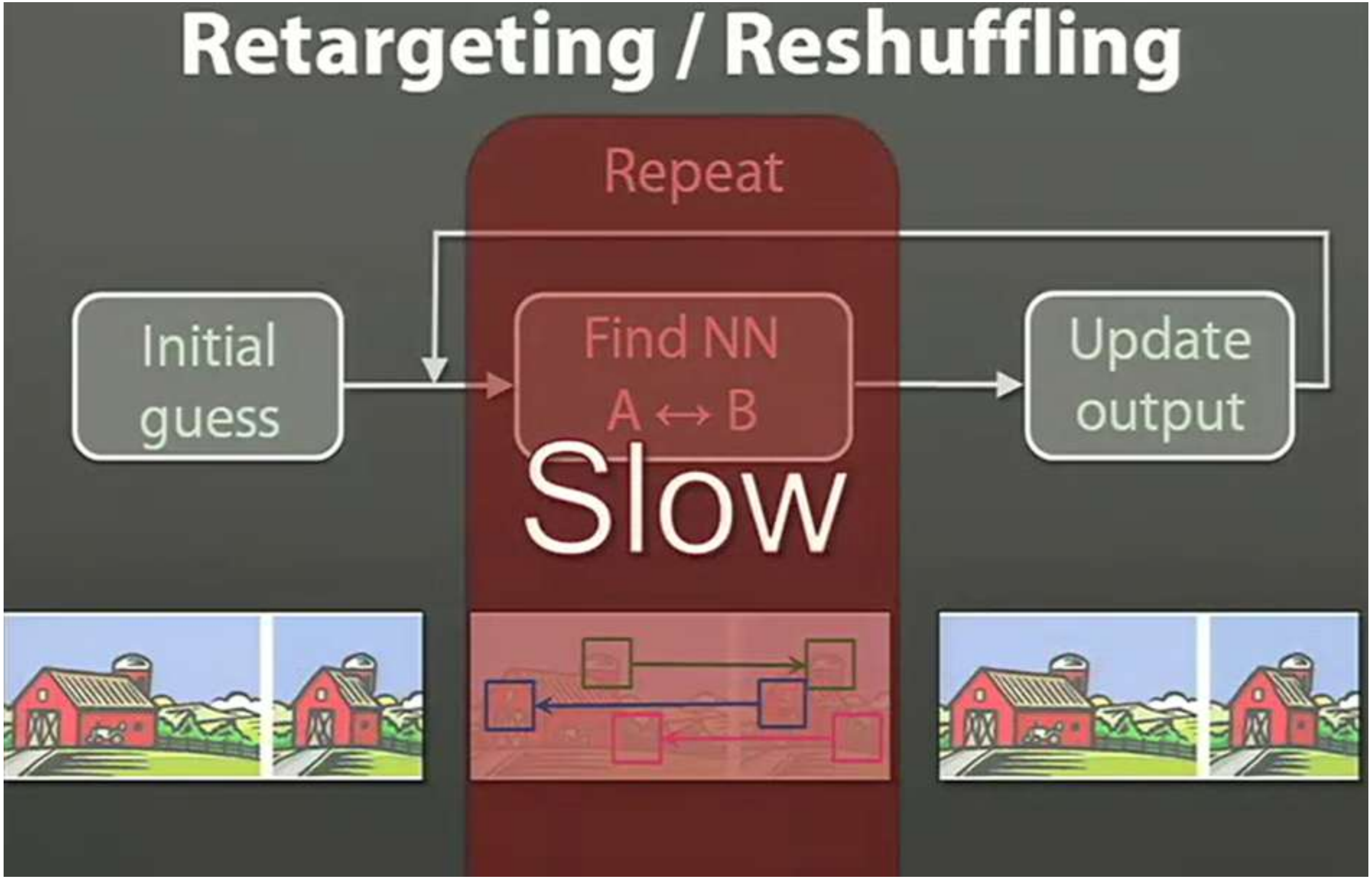
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



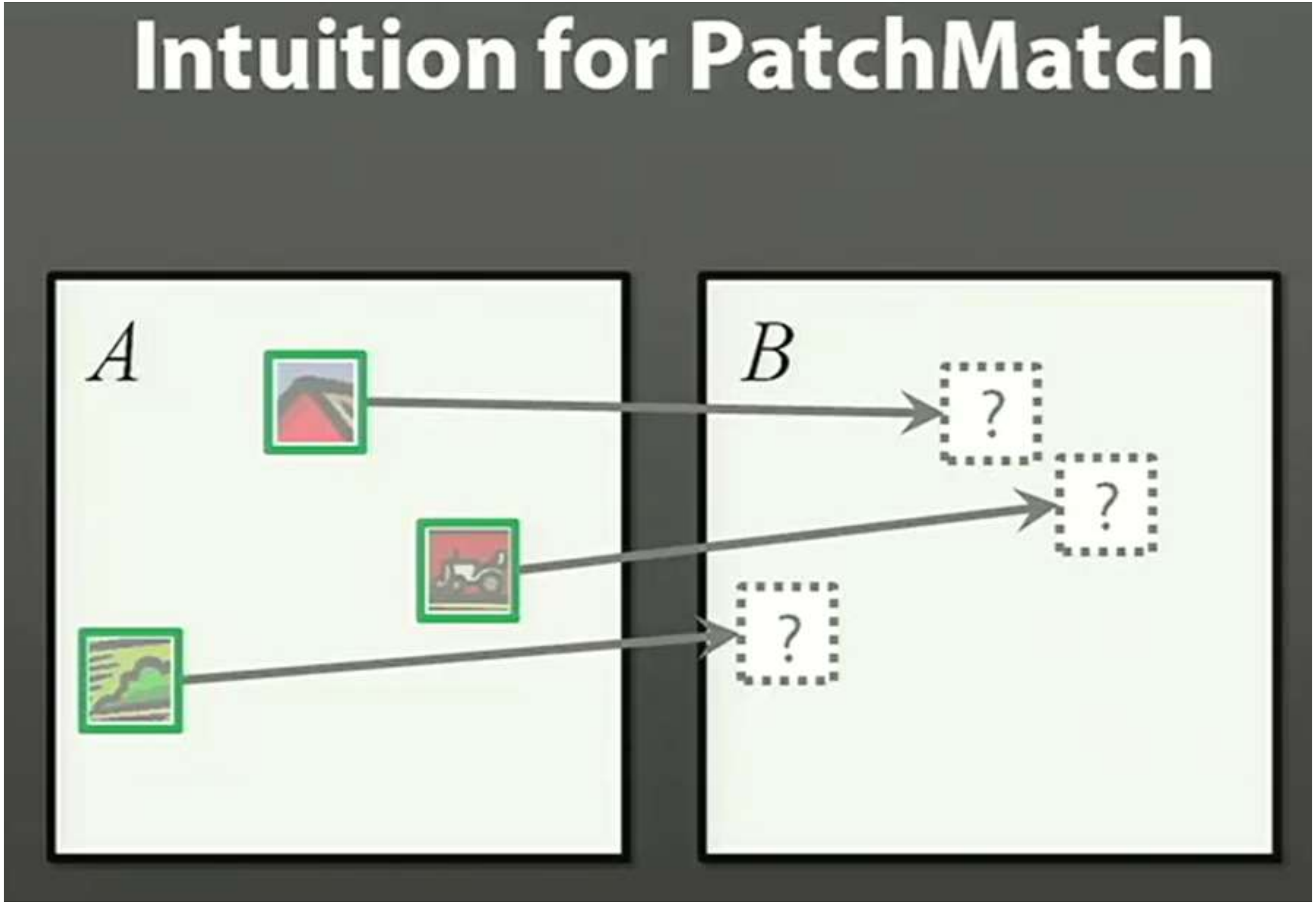
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



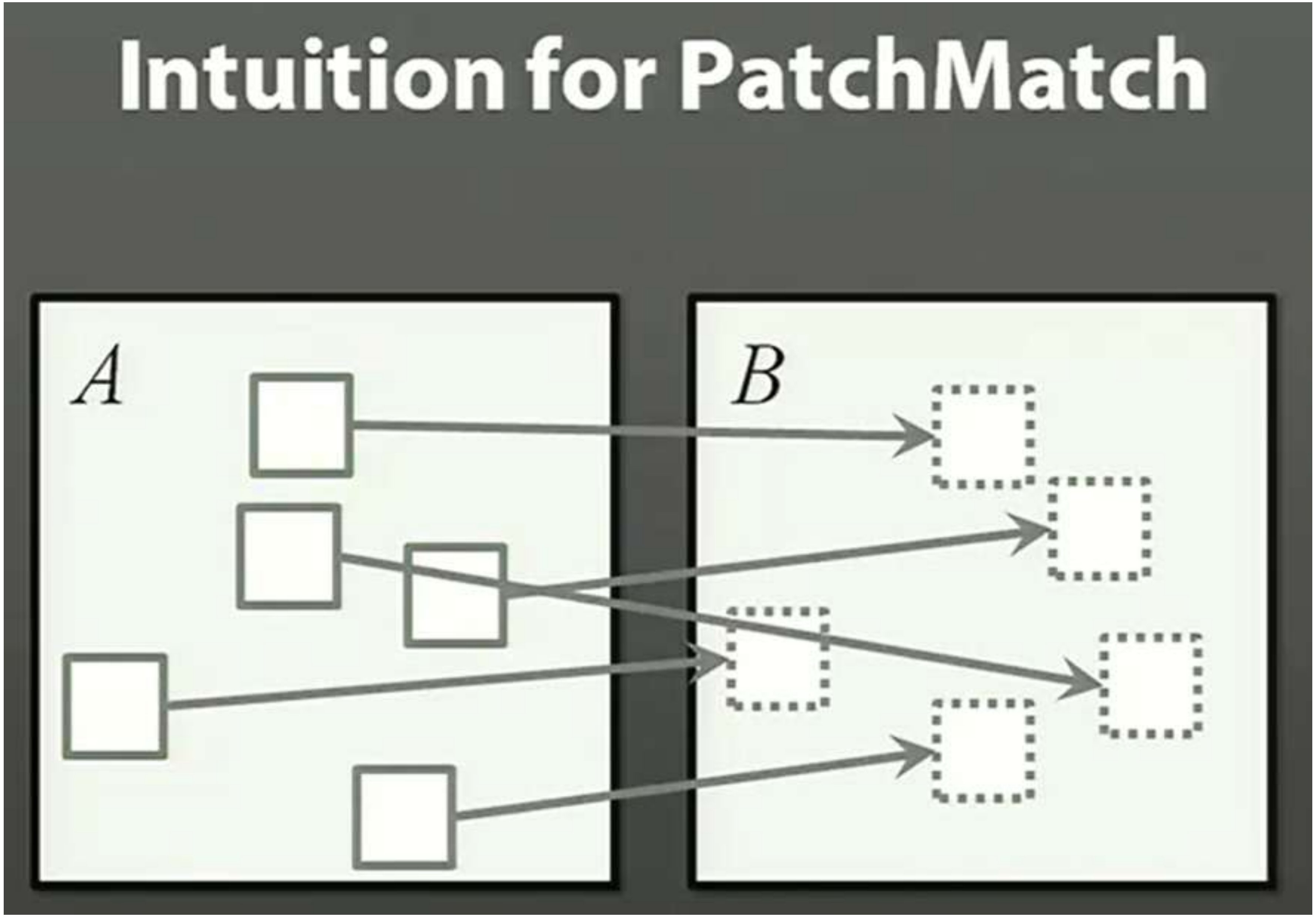
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



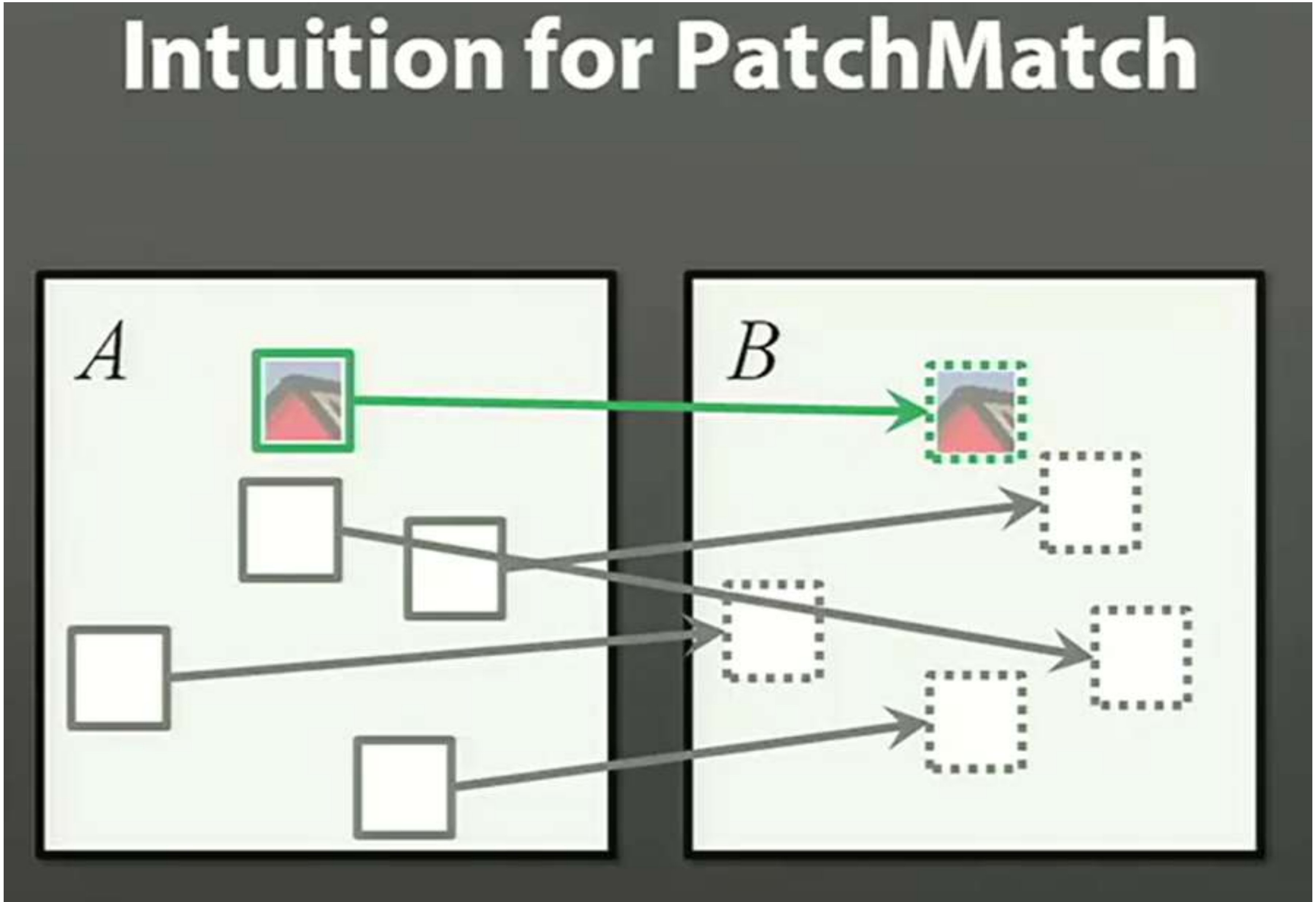
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



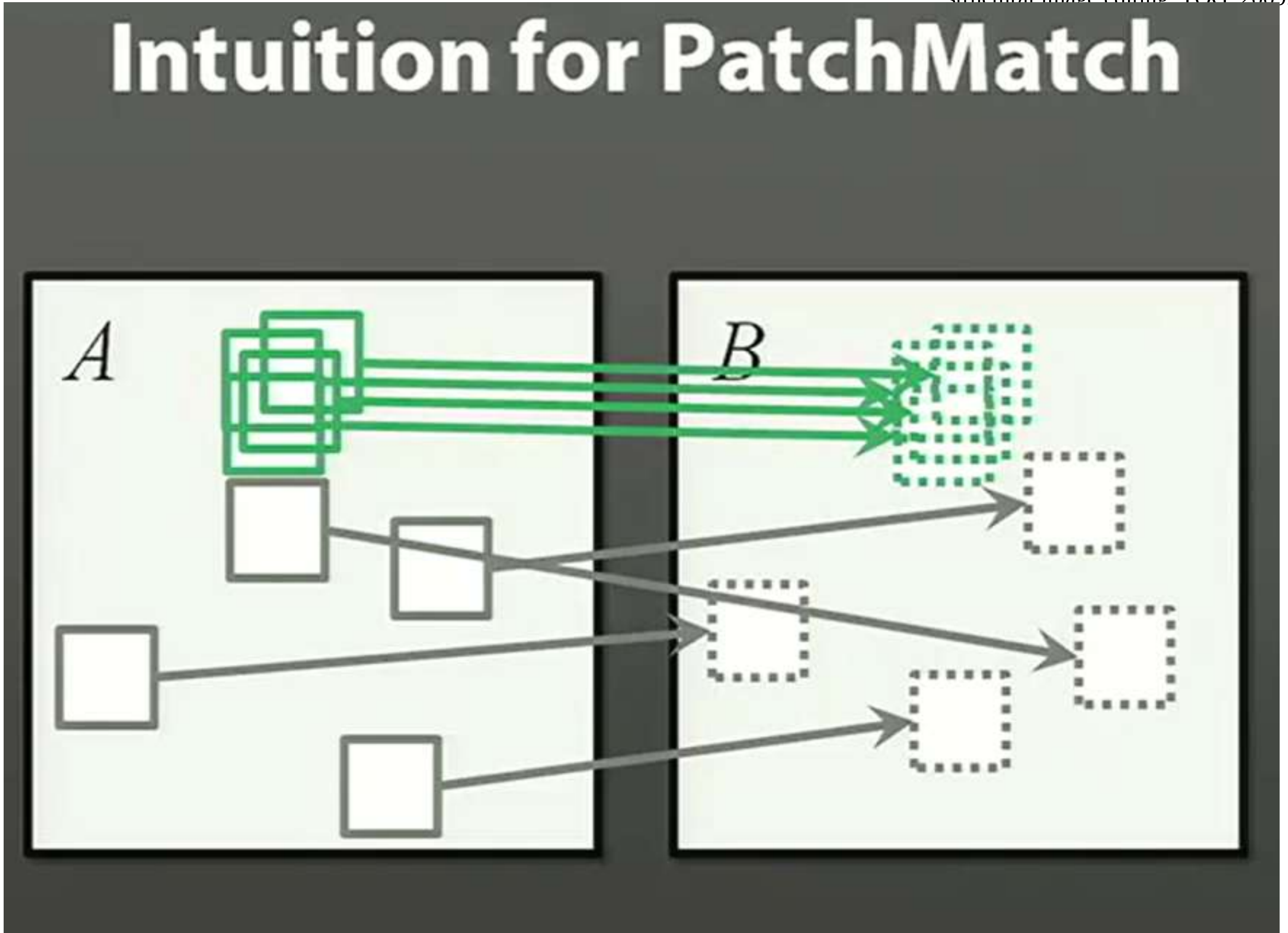
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



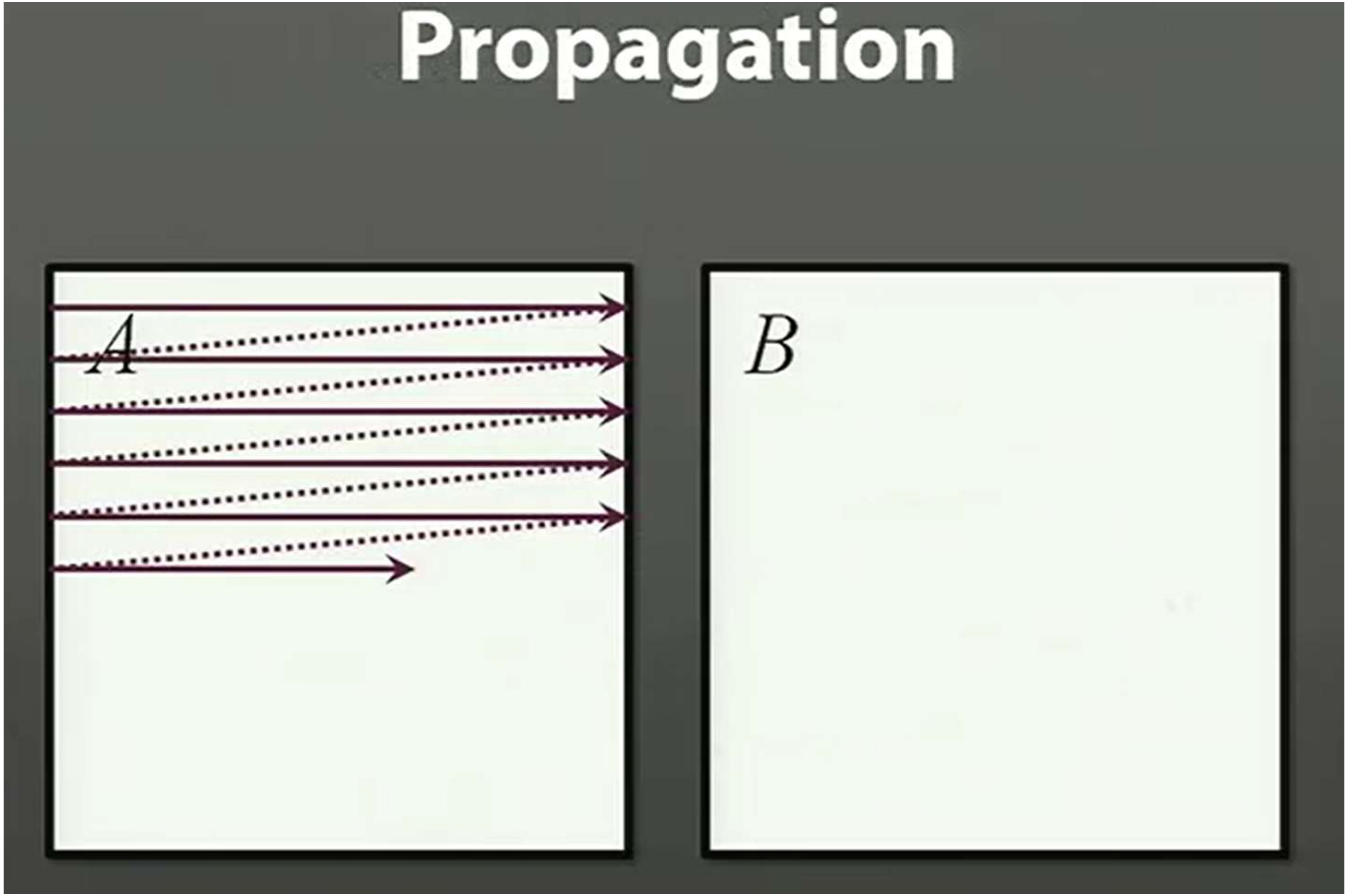
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



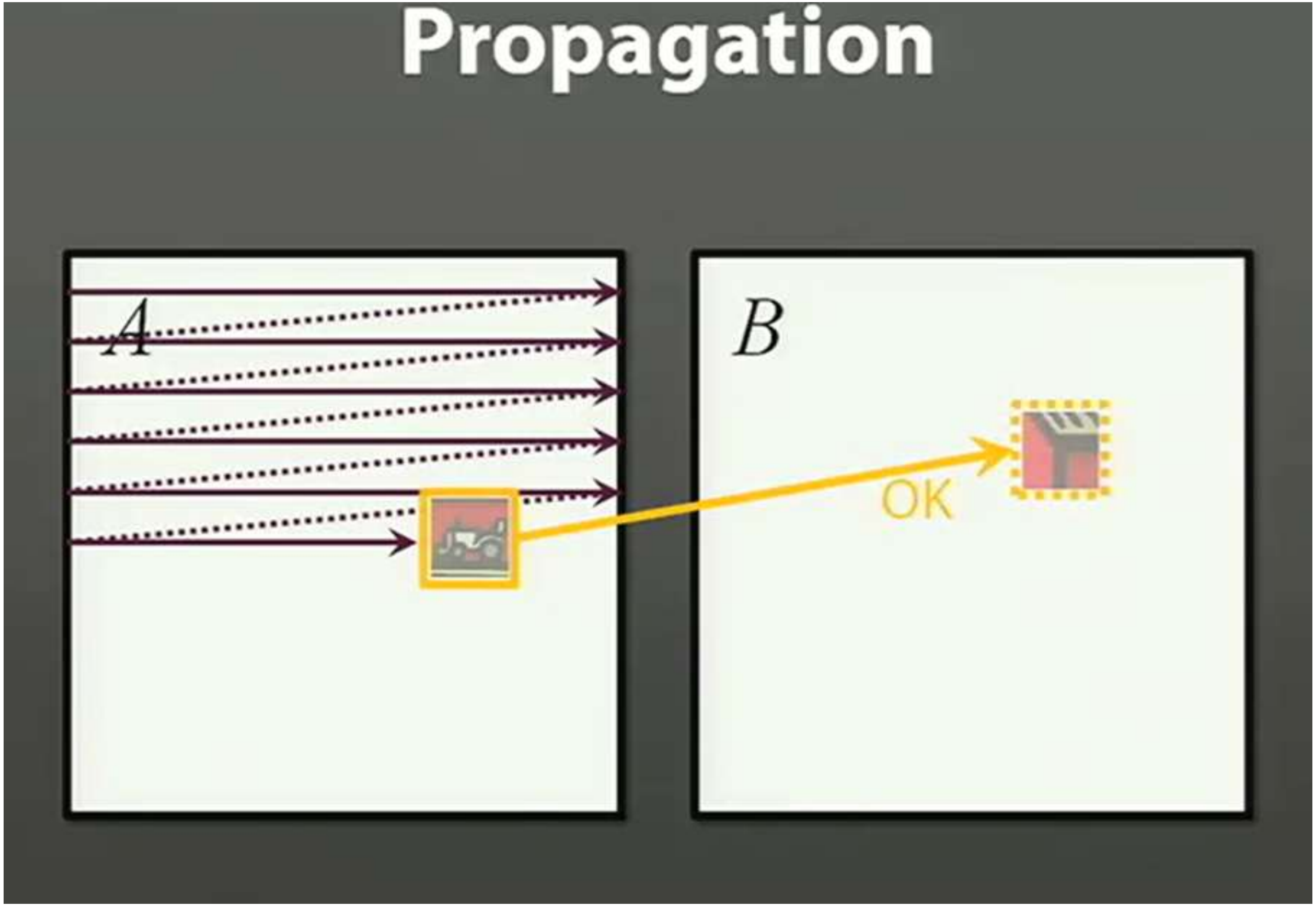
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



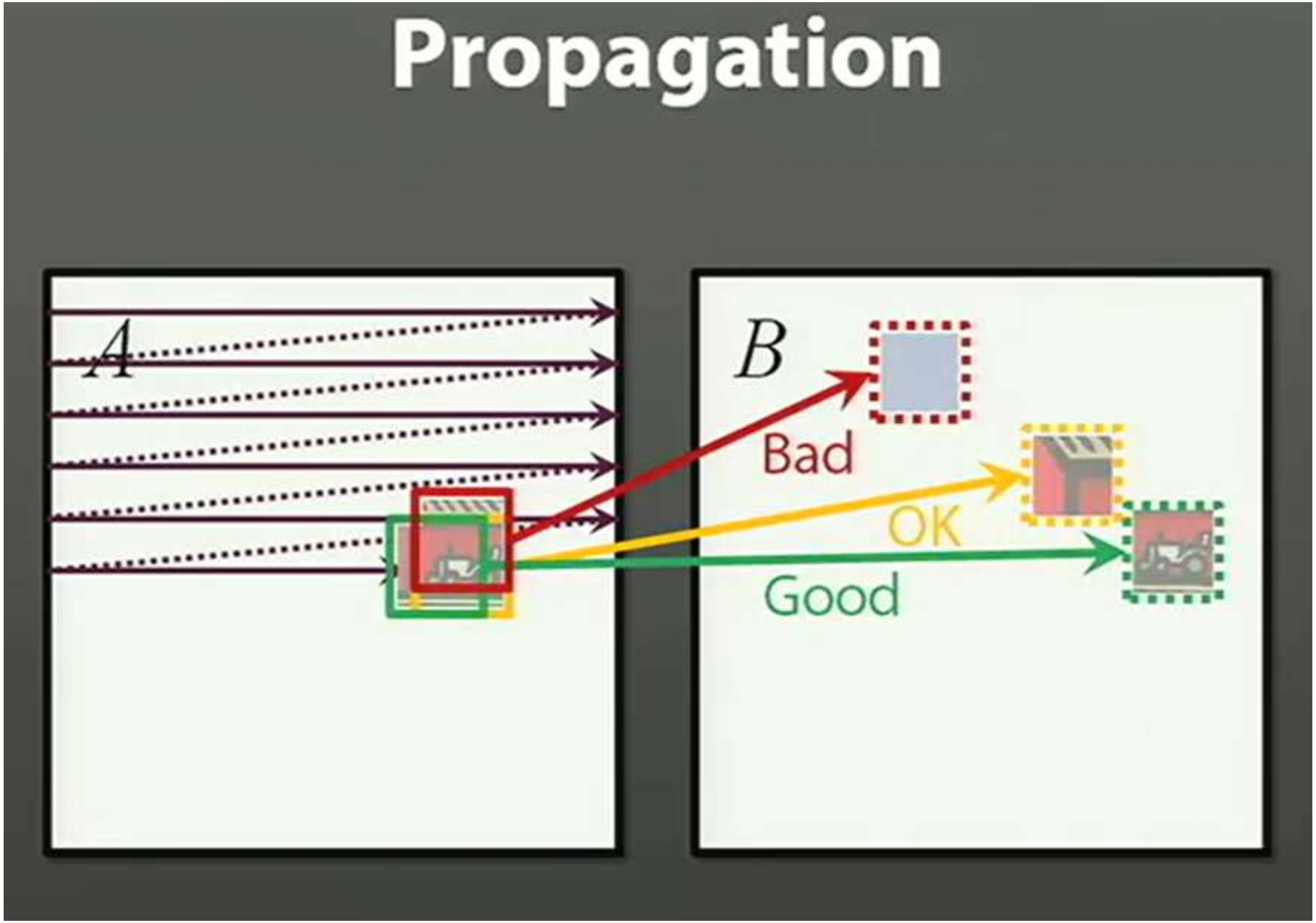
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



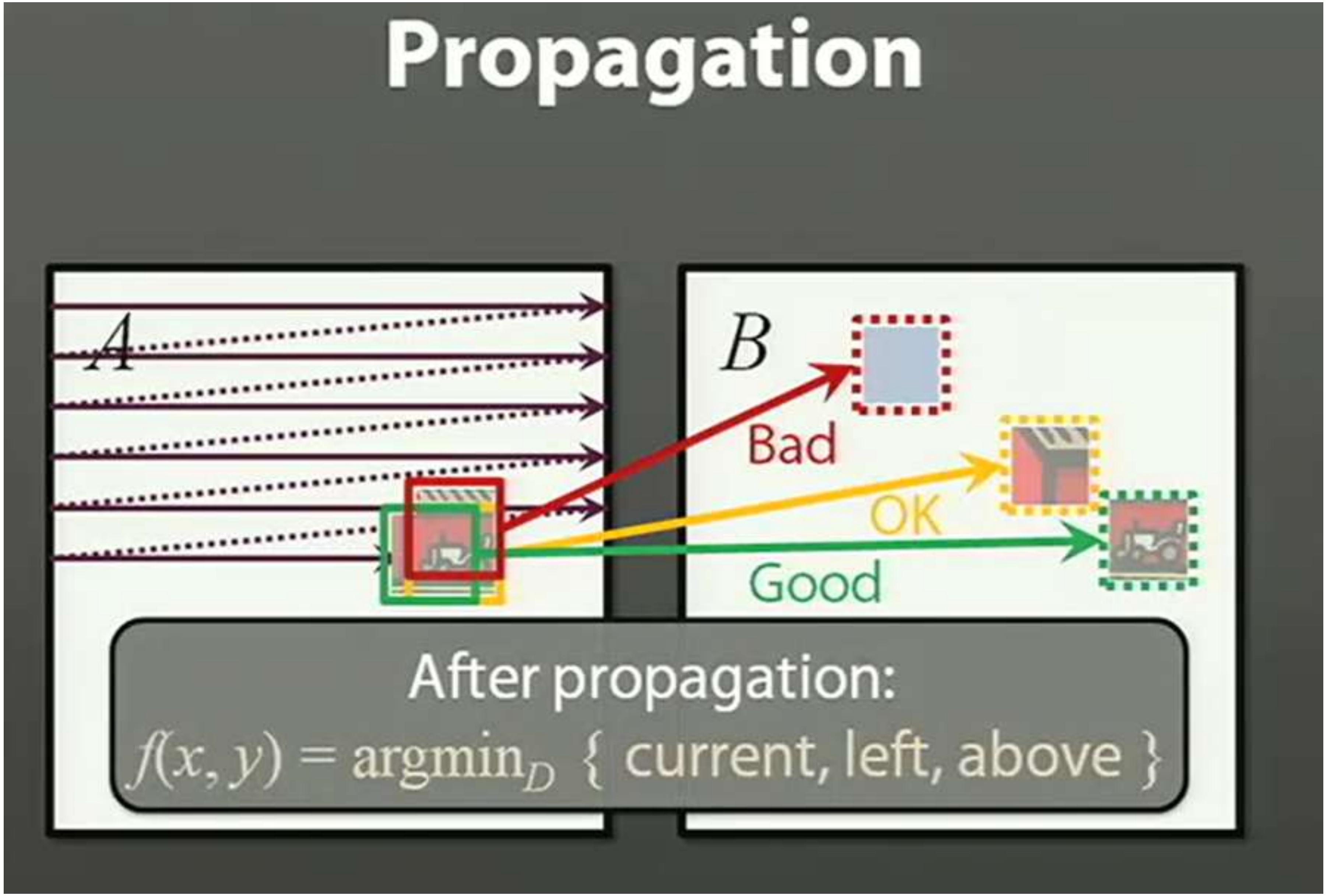
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



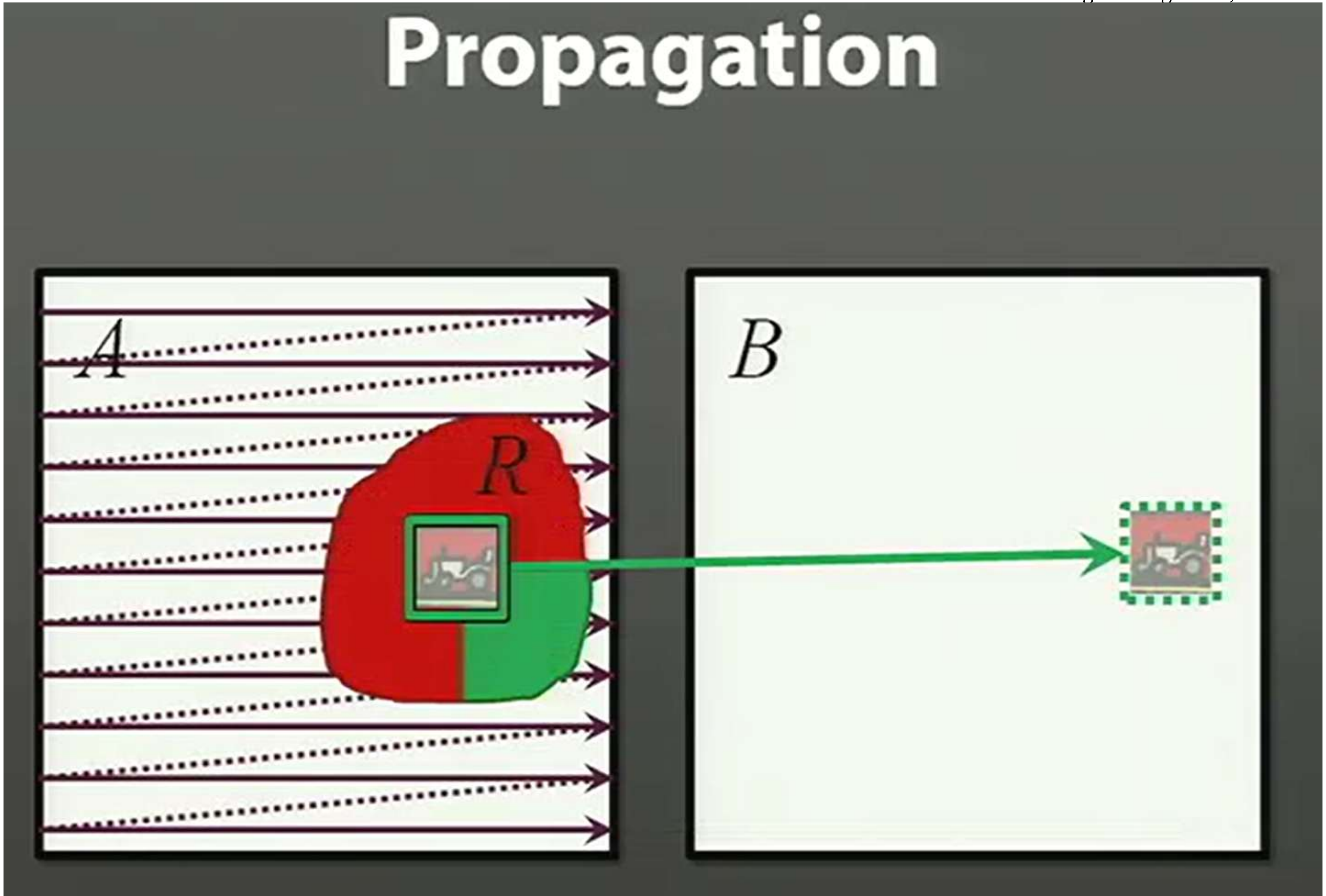
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



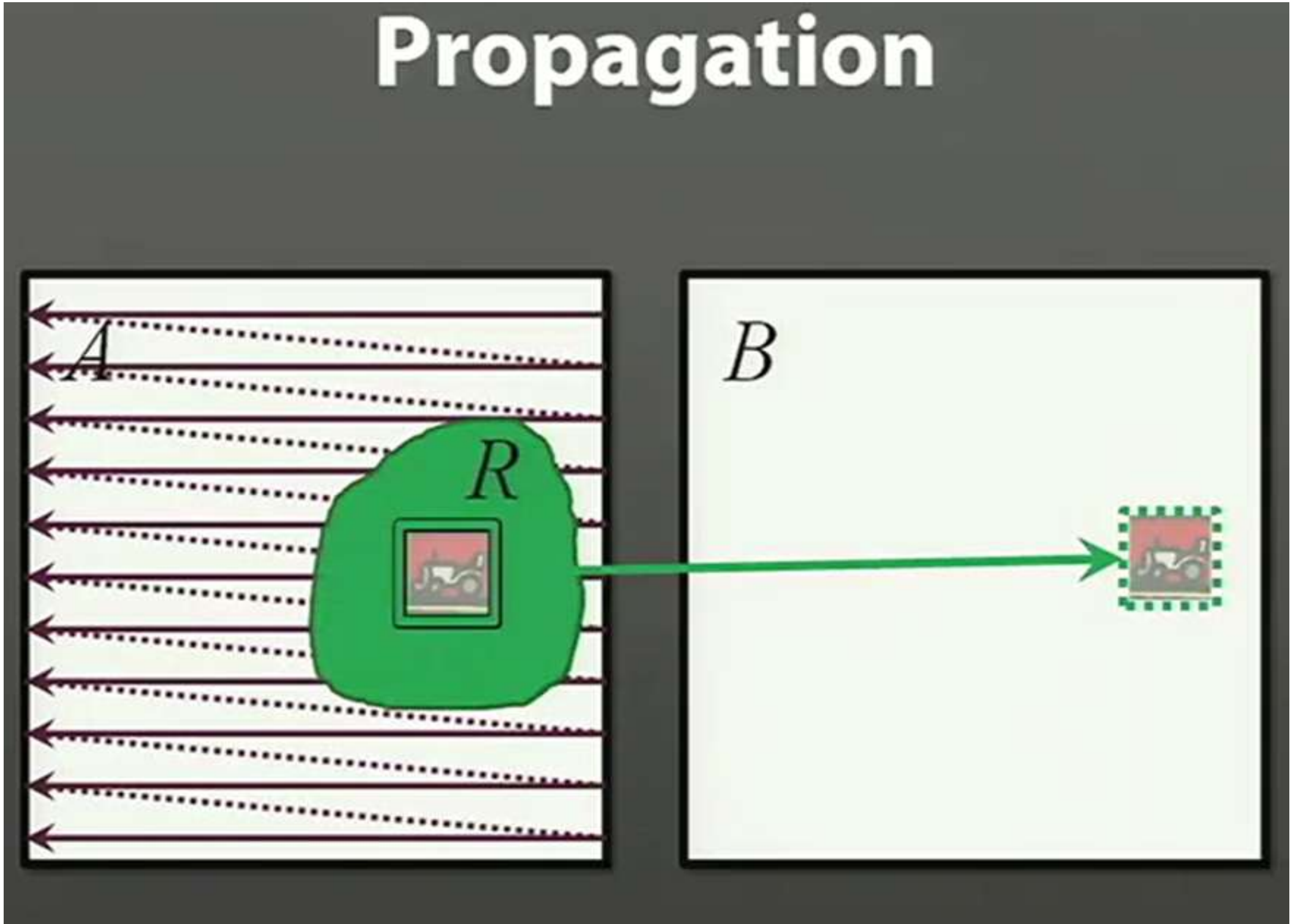
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



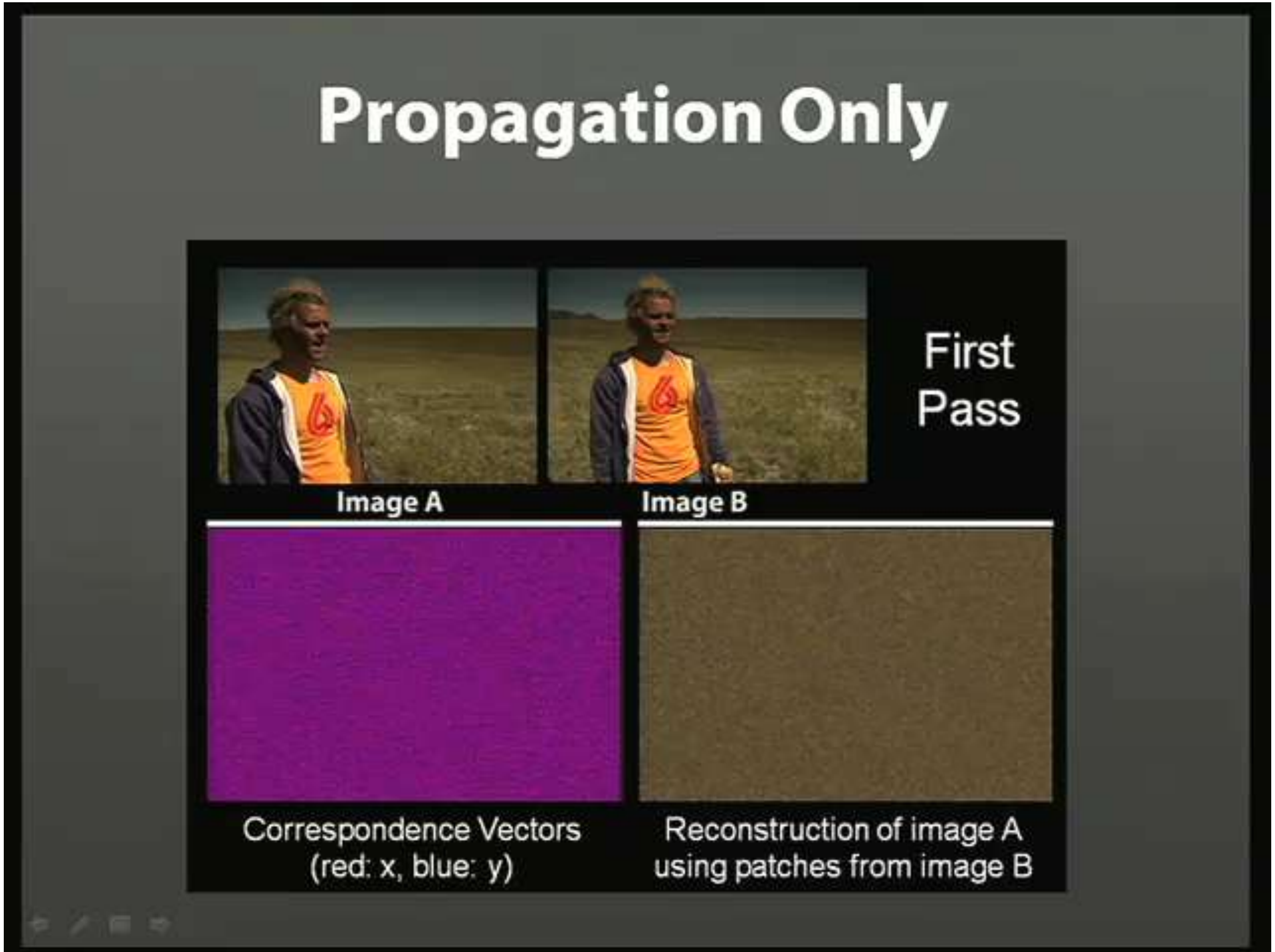
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



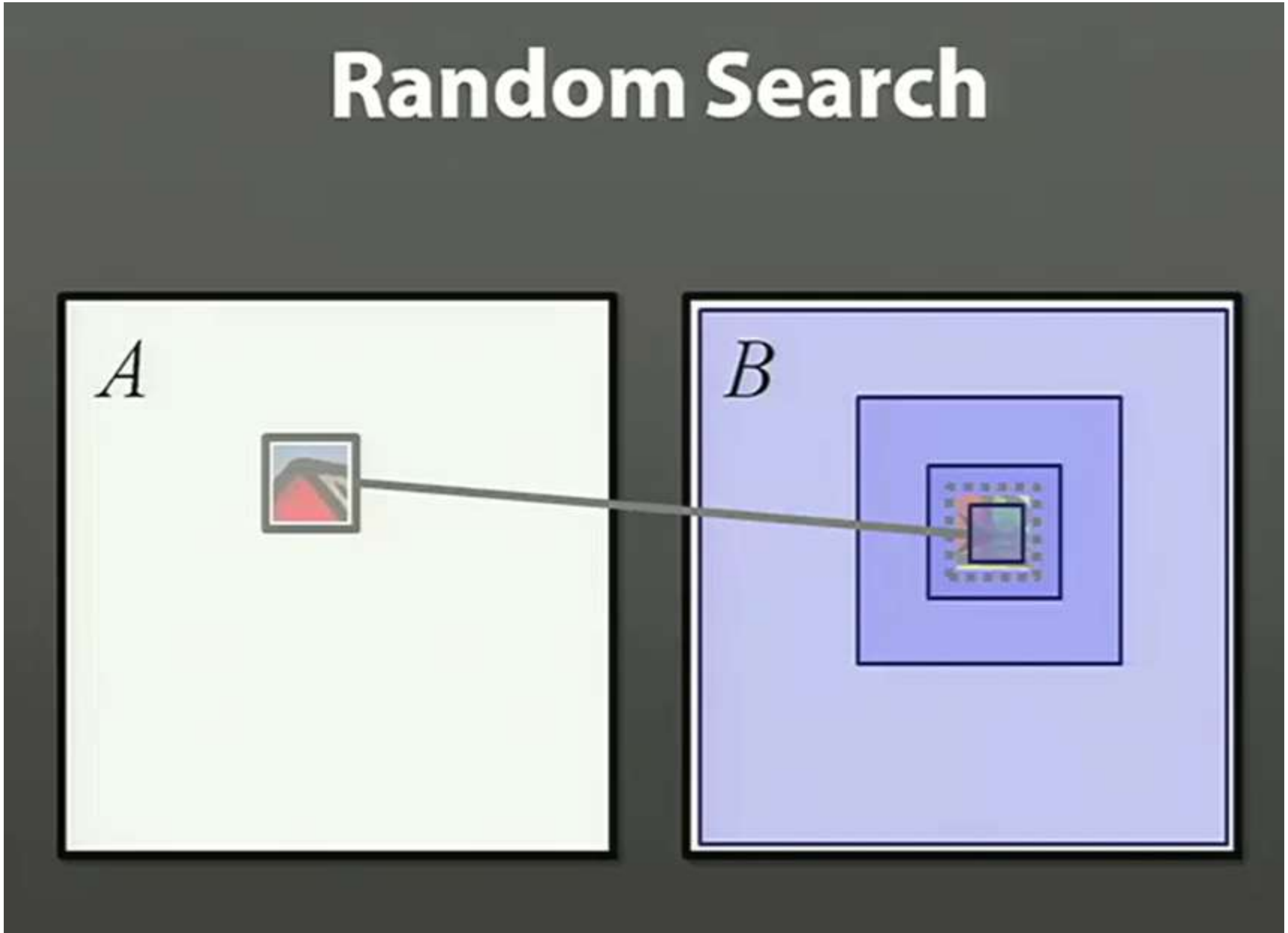
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



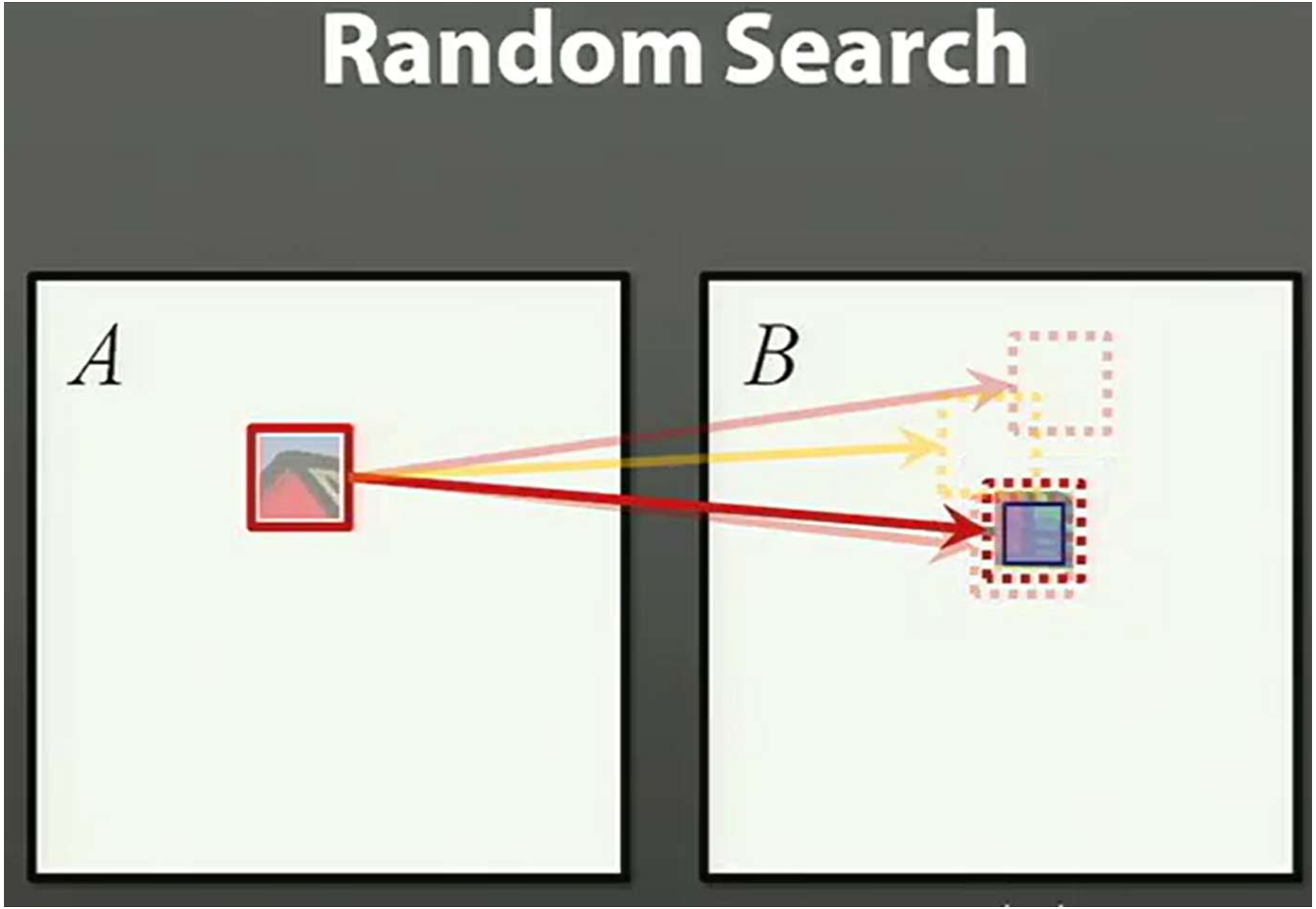
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



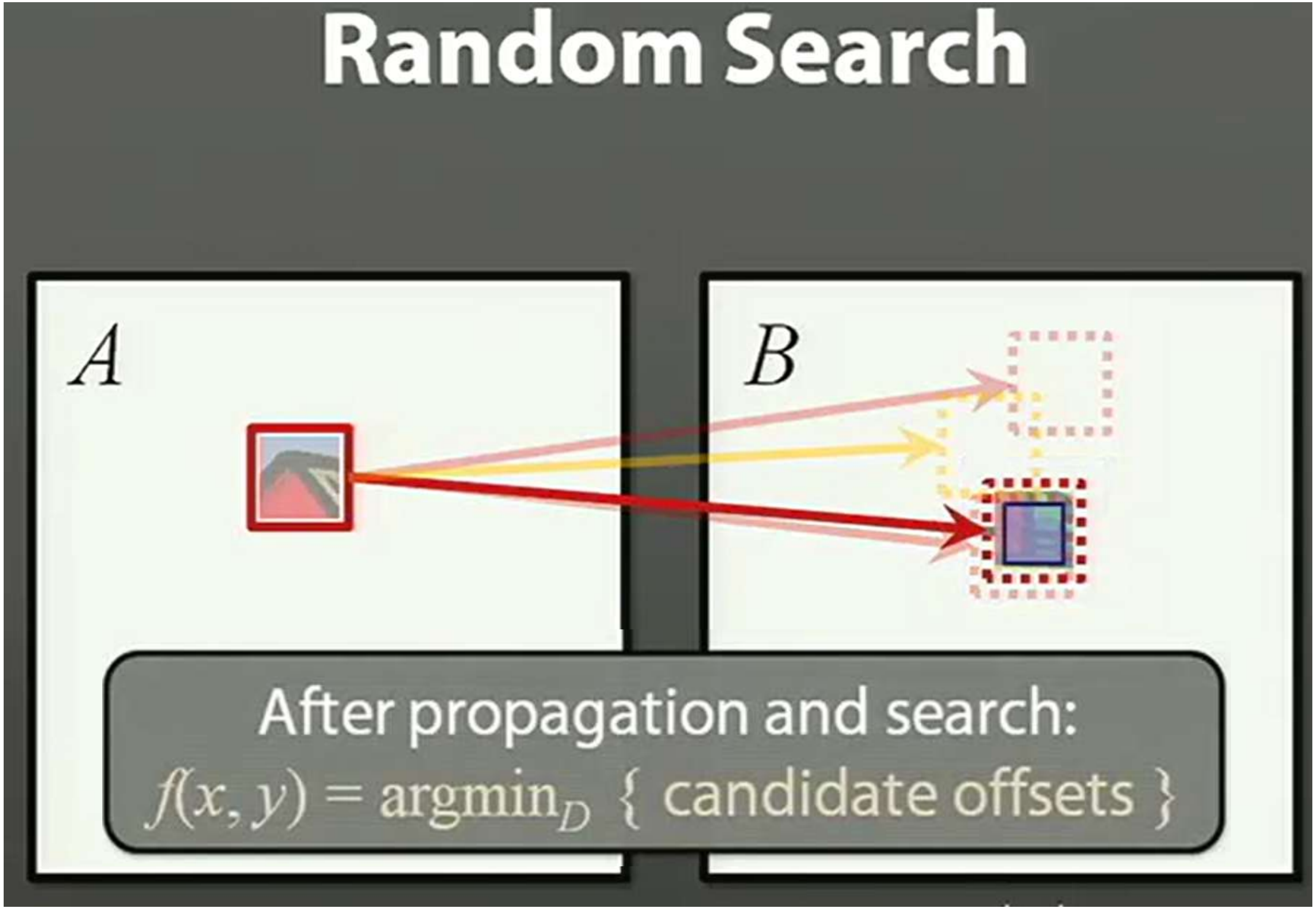
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



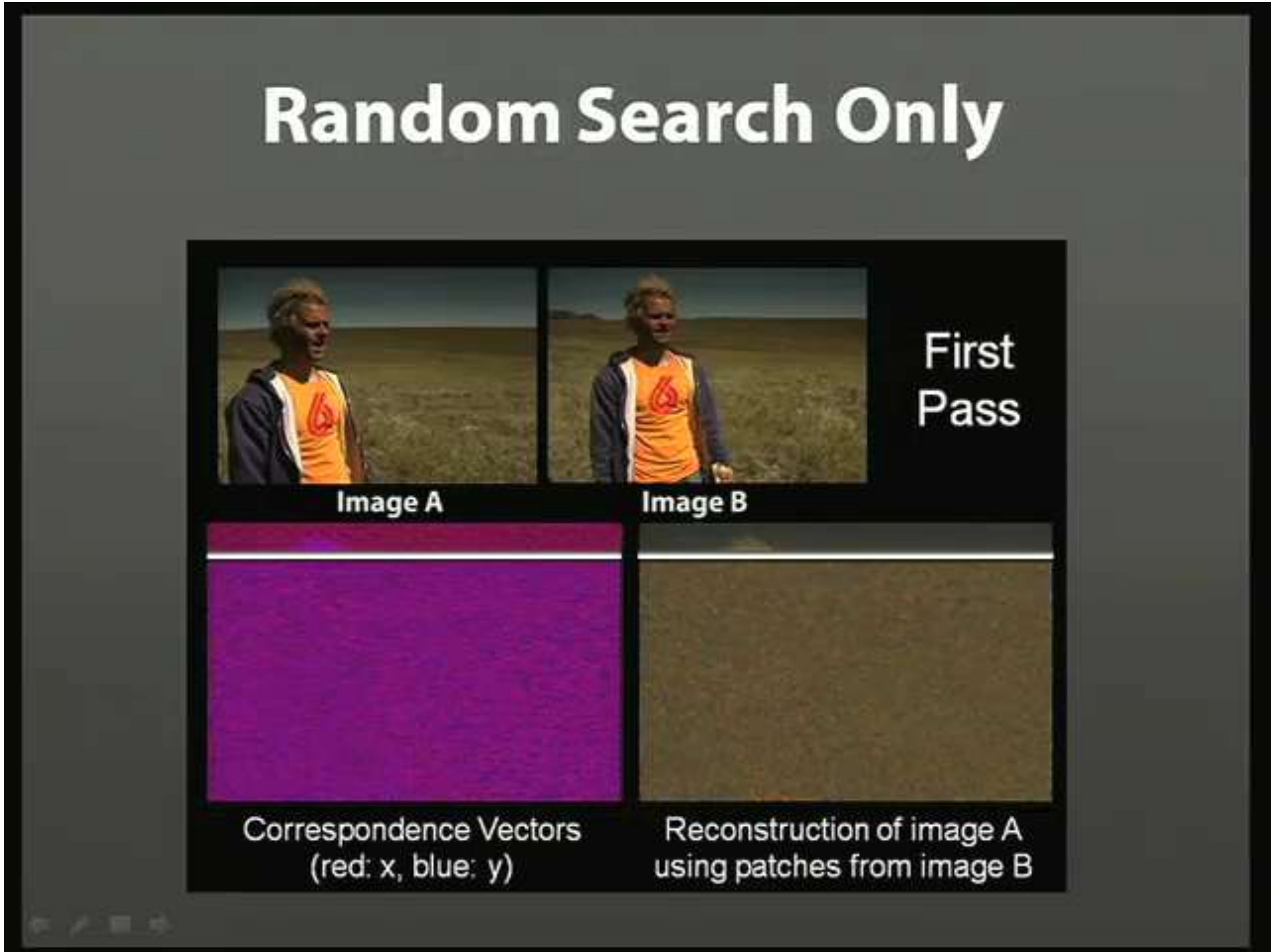
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



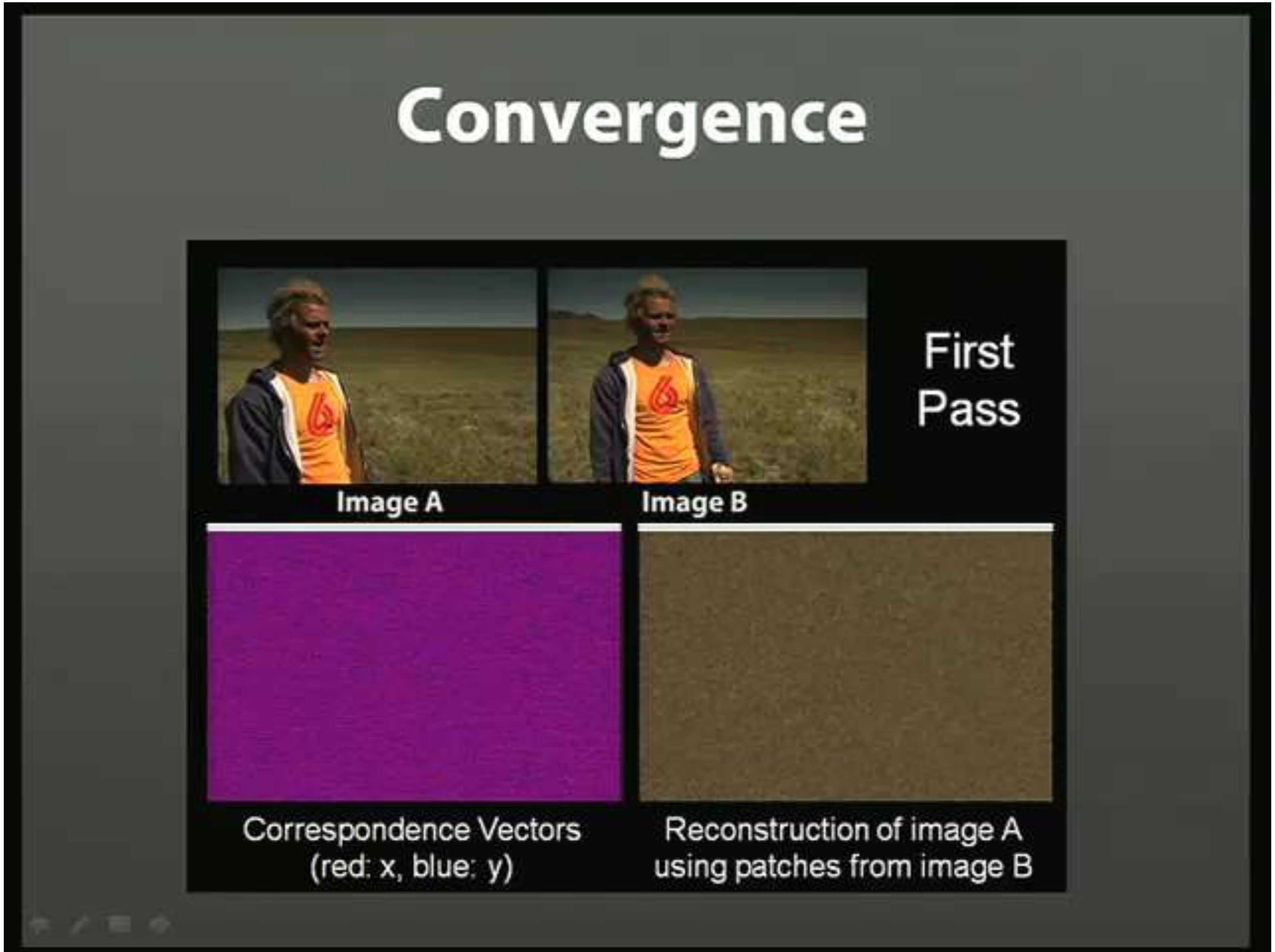
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



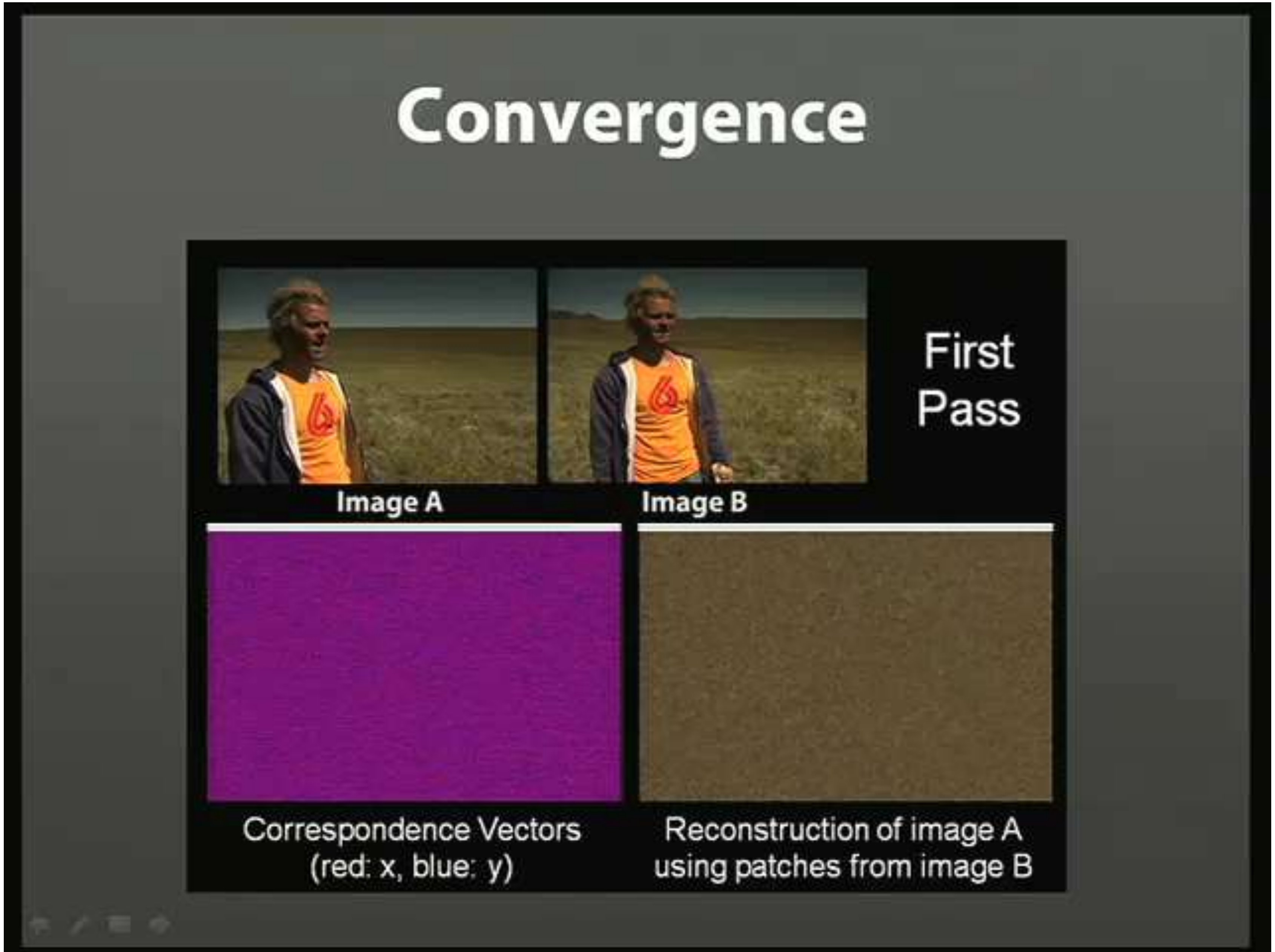
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



PatchMatch

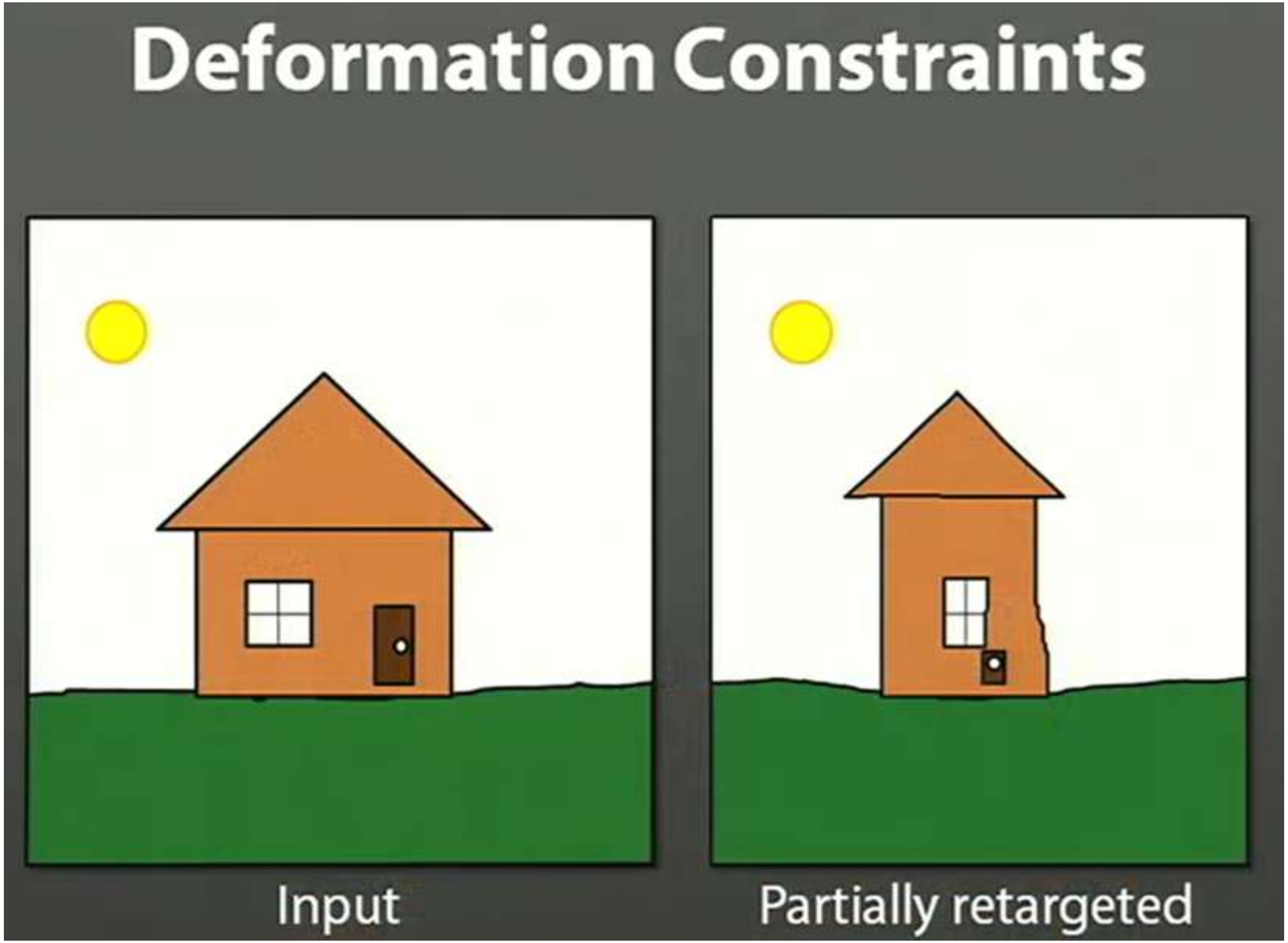
* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.

PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.

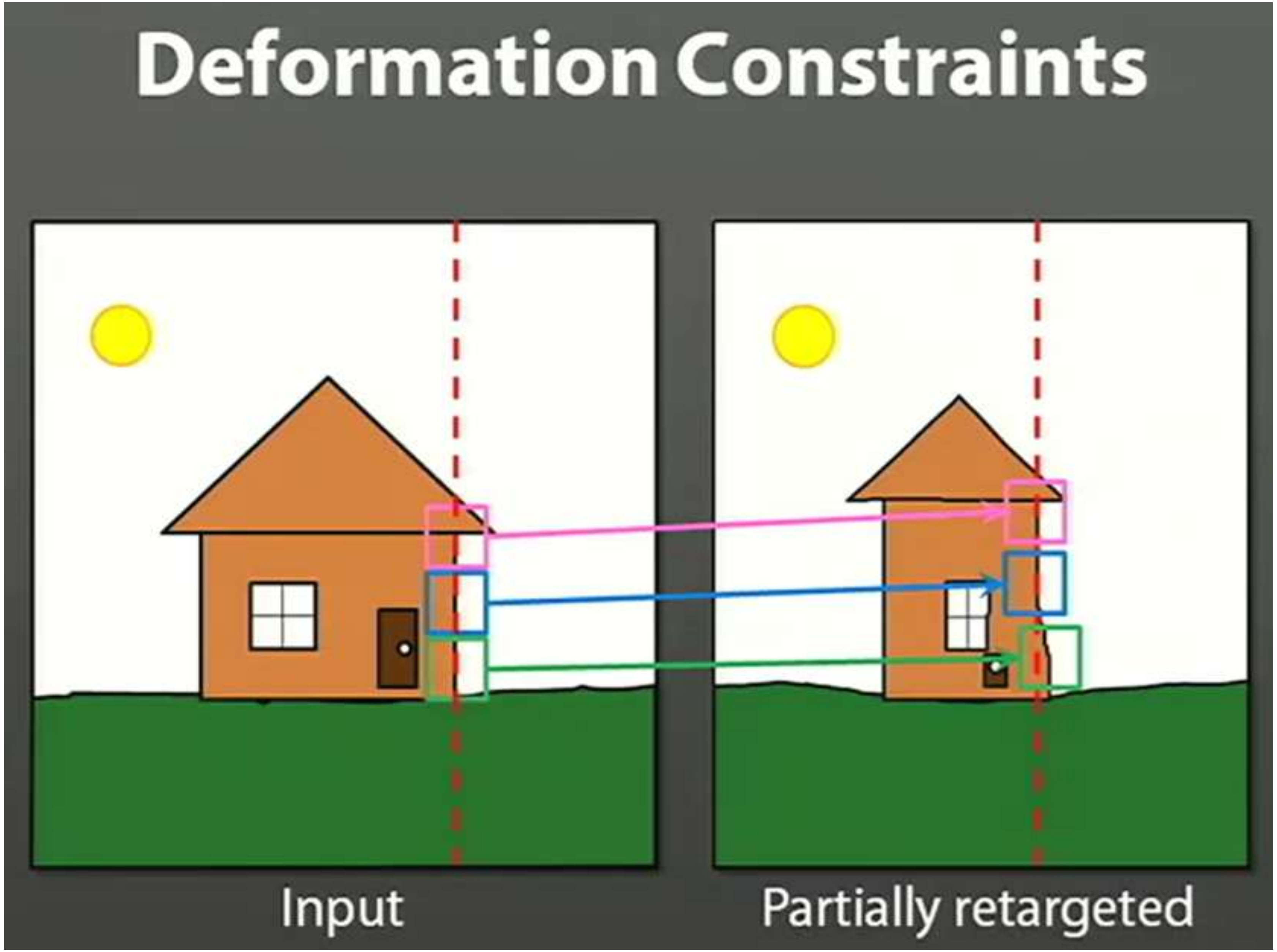
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



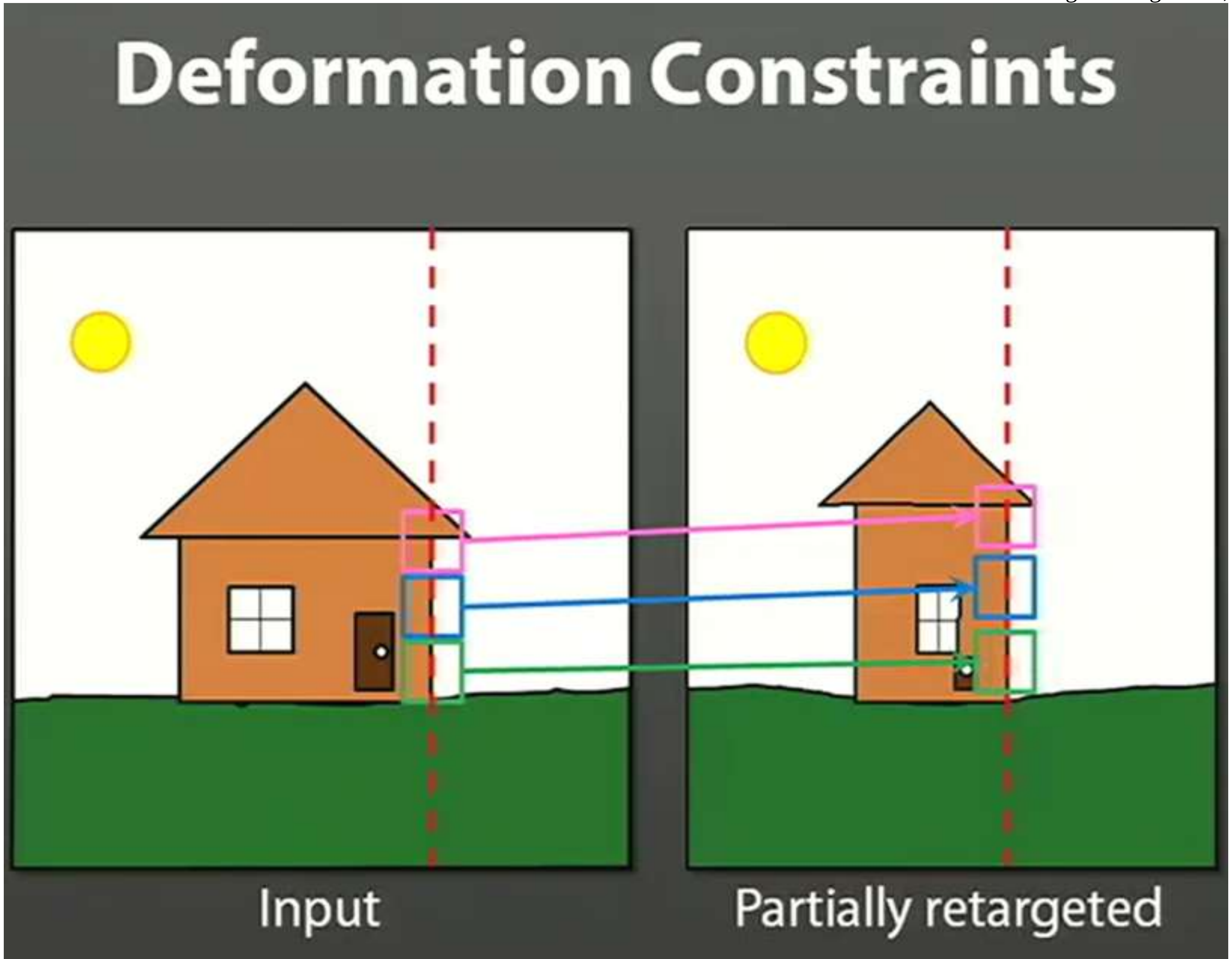
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



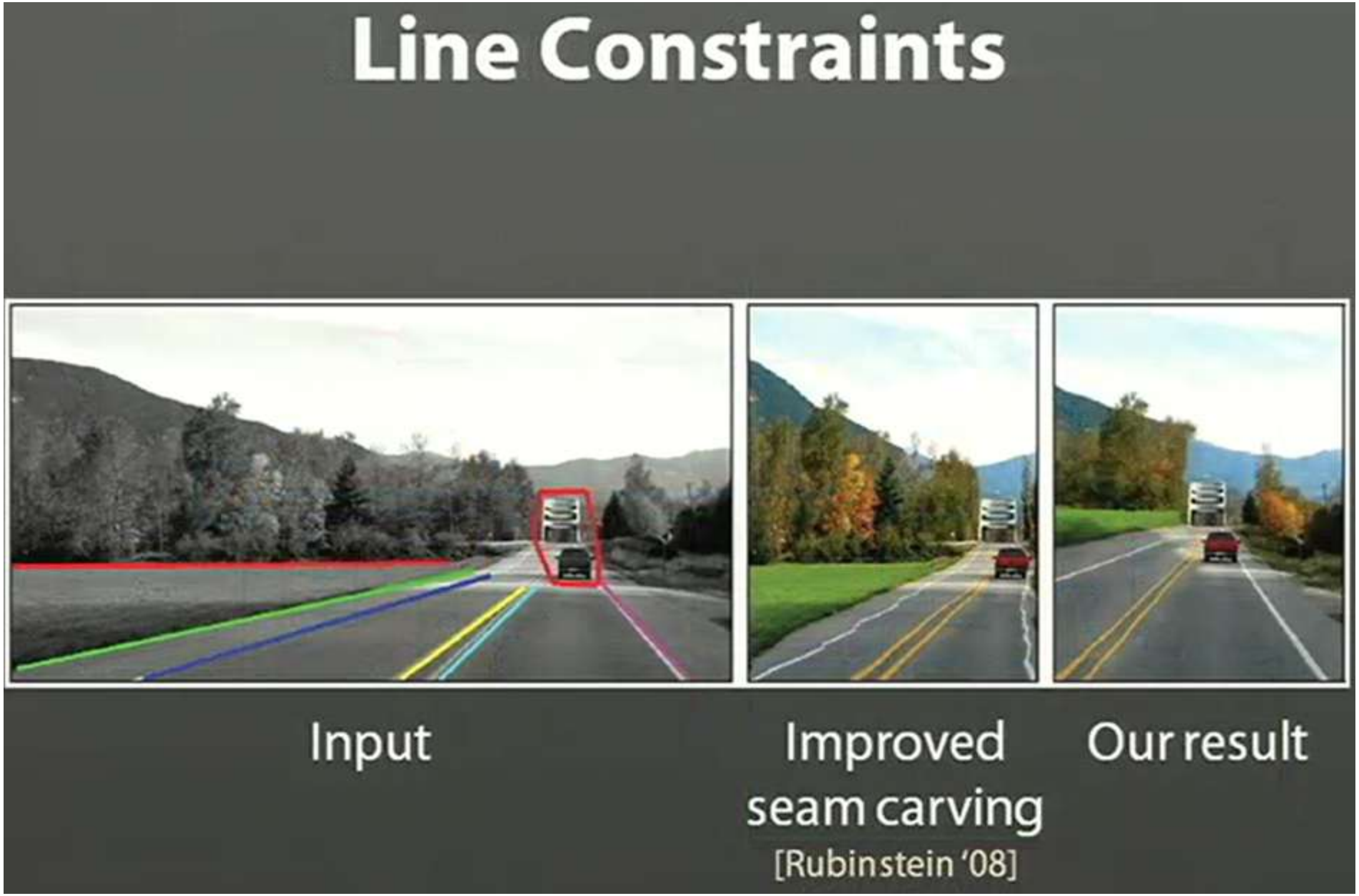
PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.

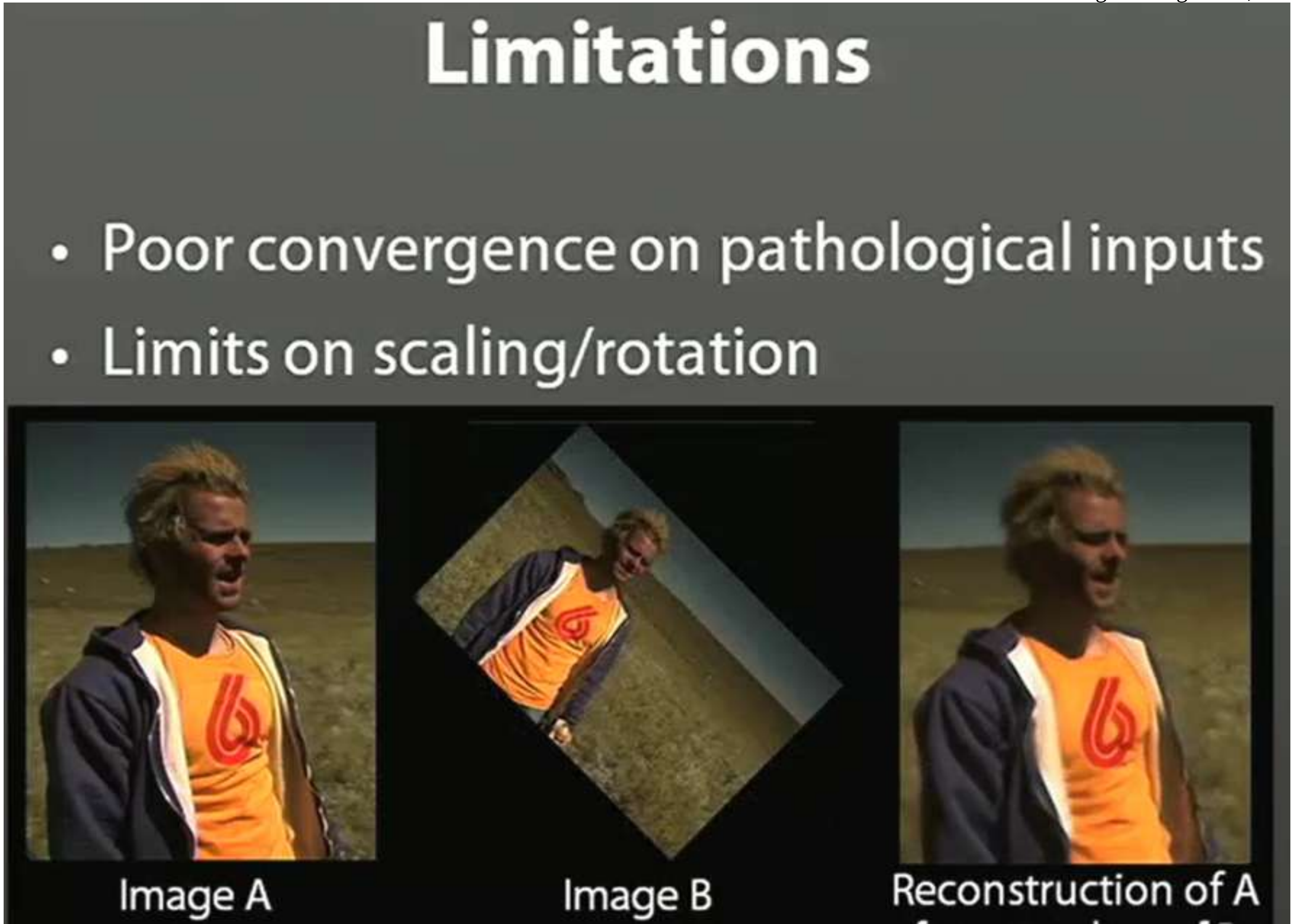


PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.

PatchMatch

* C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. TOG, 2009.



PatchMatch

- SoTA image editing 10 years ago

- We can see some metaphors…
  - Propagation: CNN, Autoregressive Models
  - Random Search：Diffusion Models, Transformer
  - ControlNet

- But there are no new content generated…
  - Just algorithms, not models (without learning)

Deep Image Prior

Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

In image restoration problems the goal is to recover original image x having a corrupted image x_0. Such problems are often formulated as an optimization task:

$$\min_x E(x; x_0) + R(x) \,, \tag{1}$$

where $E(x; x_0)$ is a *data term* and $R(x)$ is an *image prior*. The data term $E(x; x_0)$ is usually easy to design for a wide range of problems, such as super-resolution, denoising, inpainting, while image prior $R(x)$ is a challenging one. Today's trend is to capture the prior $R(x)$ with a ConvNet by training it using large number of examples.

We first notice, that for a surjective $g : \theta \mapsto x$ the following procedure in theory is equivalent to (1):

$$\min_\theta E(g(\theta); x_0) + R(g(\theta)) \,.$$

In practice g dramatically changes how the image space is searched by an optimization method. Furthermore, by selecting a "good" (possibly injective) mapping g, we could get rid of the prior term. We define $g(\theta)$ as $f_\theta(z)$, where f is a deep ConvNet with parameters θ and z is a fixed input, leading to the formulation

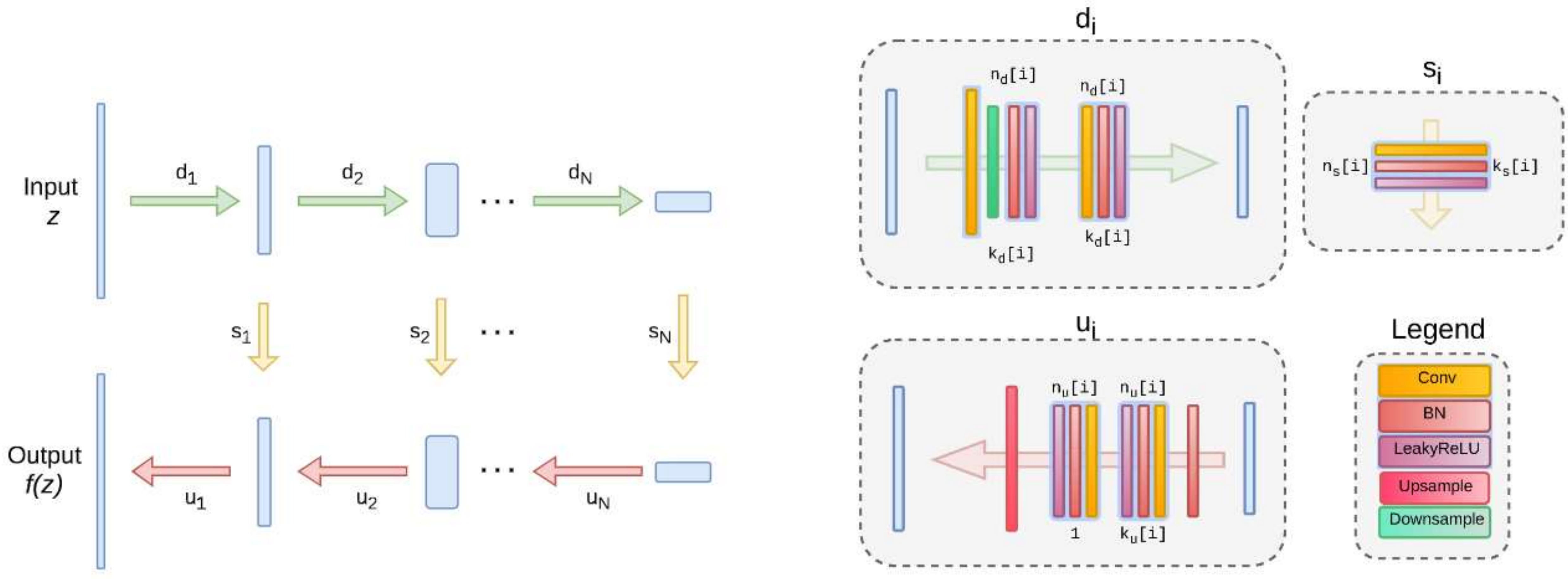
$$\min_\theta E(f_\theta(z); x_0) \,.$$

Here, the network f_θ is initialized randomly and input z is filled with noise and fixed.

In other words, **instead of searching for the answer in the image space we now search for it in the space of neural network's parameters**. We emphasize that we never use a pretrained network or an image database. Only corrupted image x_0 is used in the restoration process.

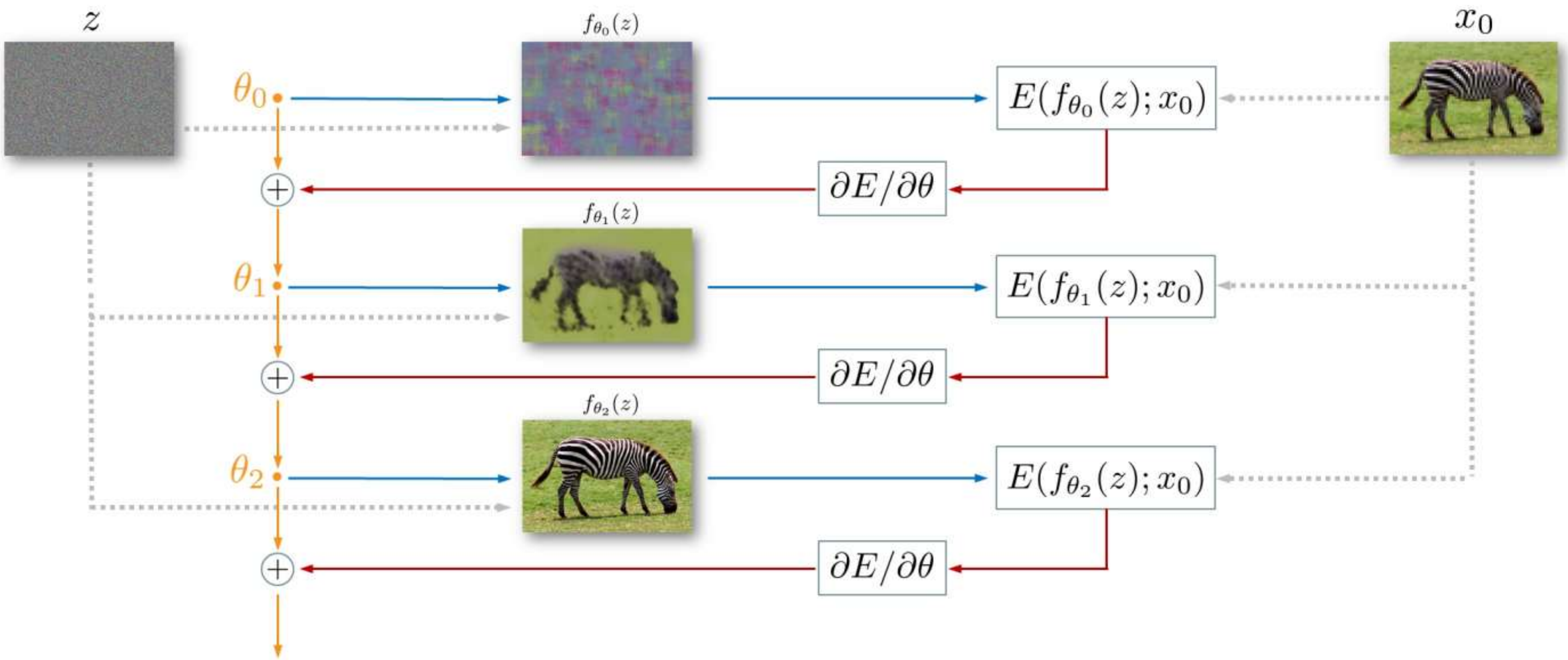
Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.



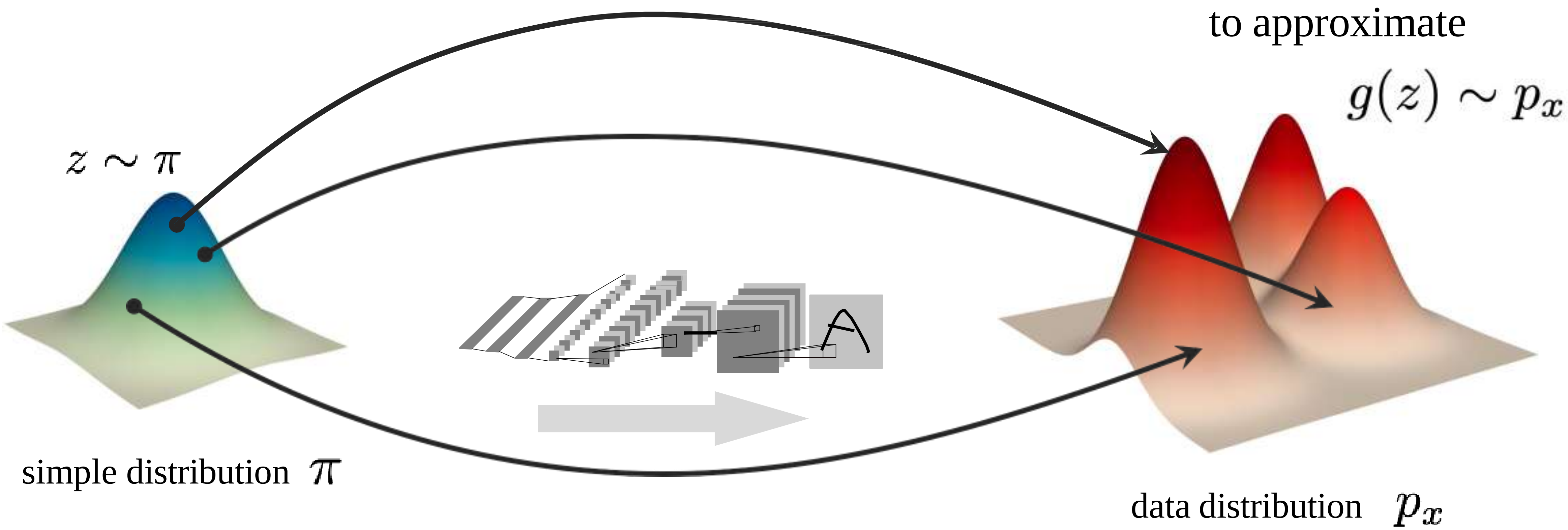
Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.



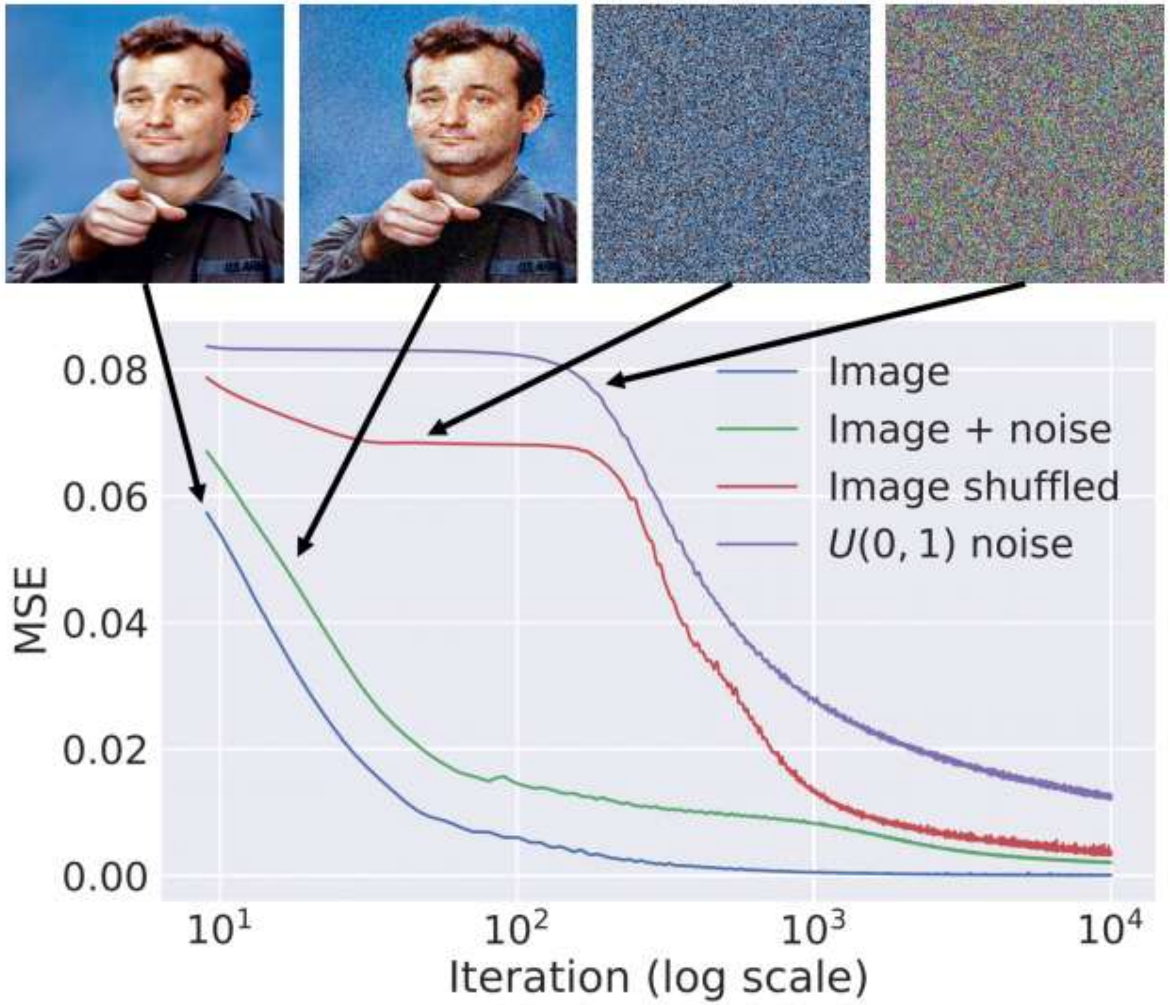
Recap.

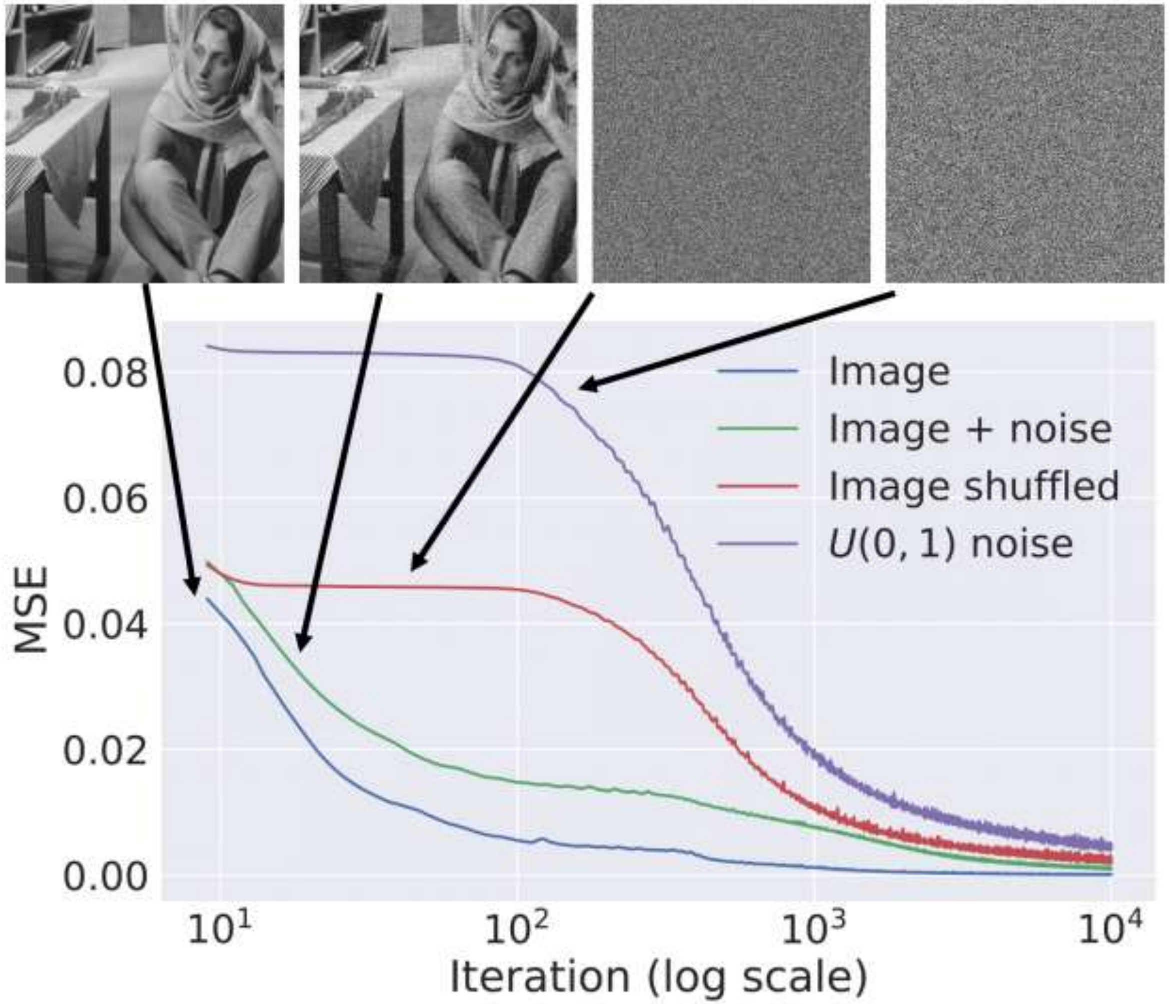
- Deep image prior uses individual network for each image



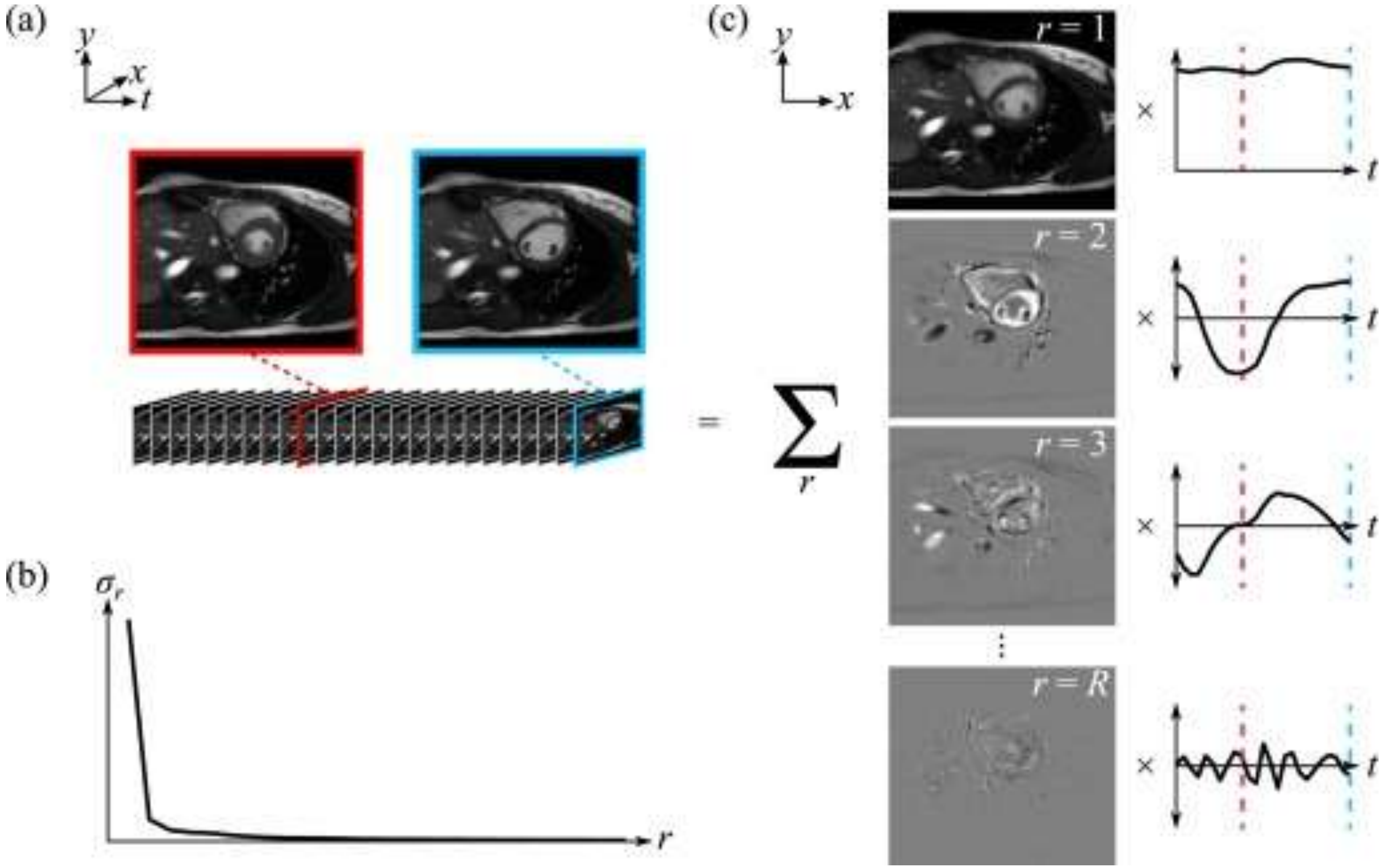
Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.





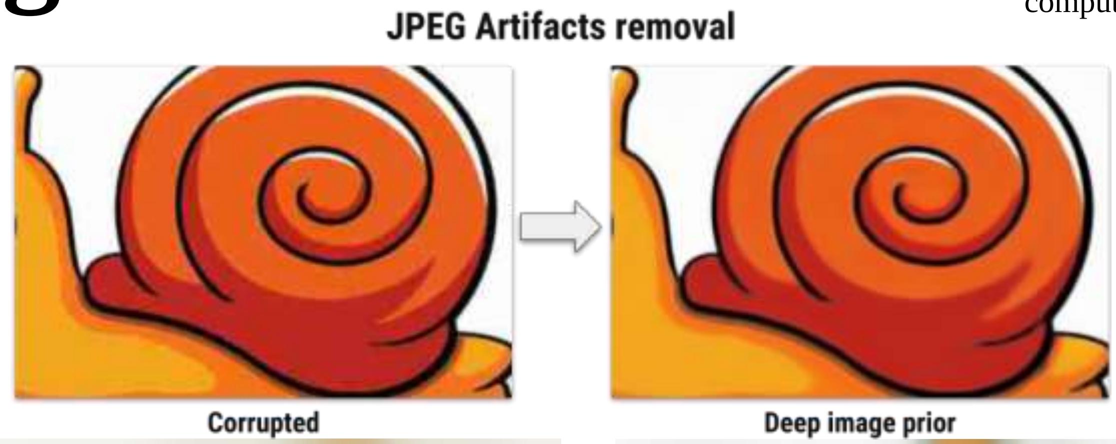
Compressed Sensing / Low-Rank Approximation

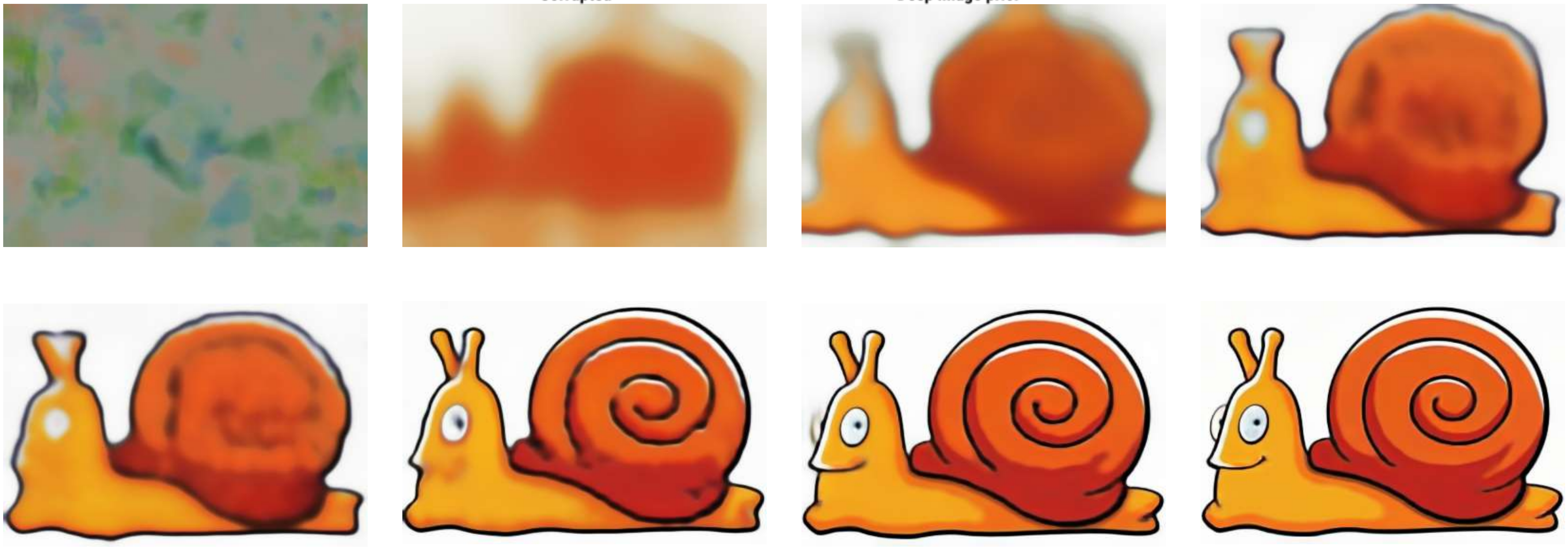


https://en.wikipedia.org/wiki/Low-rank_approximation

Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

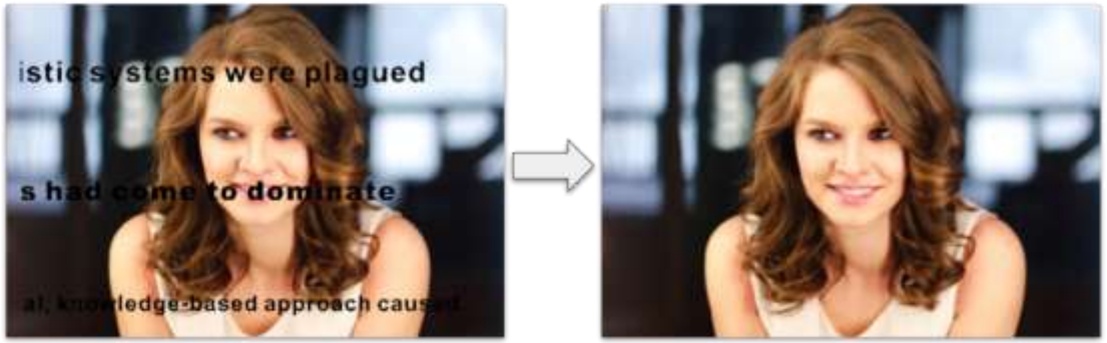




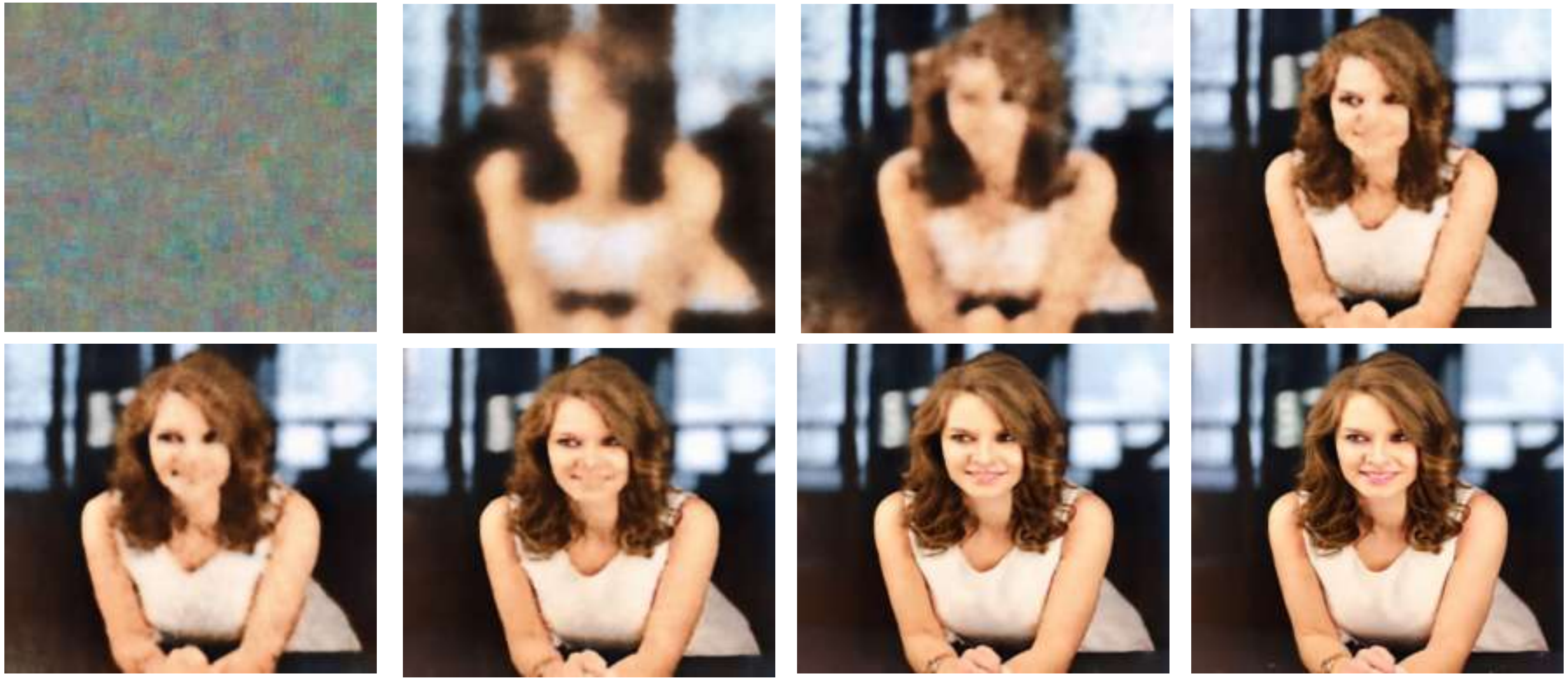
Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

Inpainting



Corrupted   Deep image prior



Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

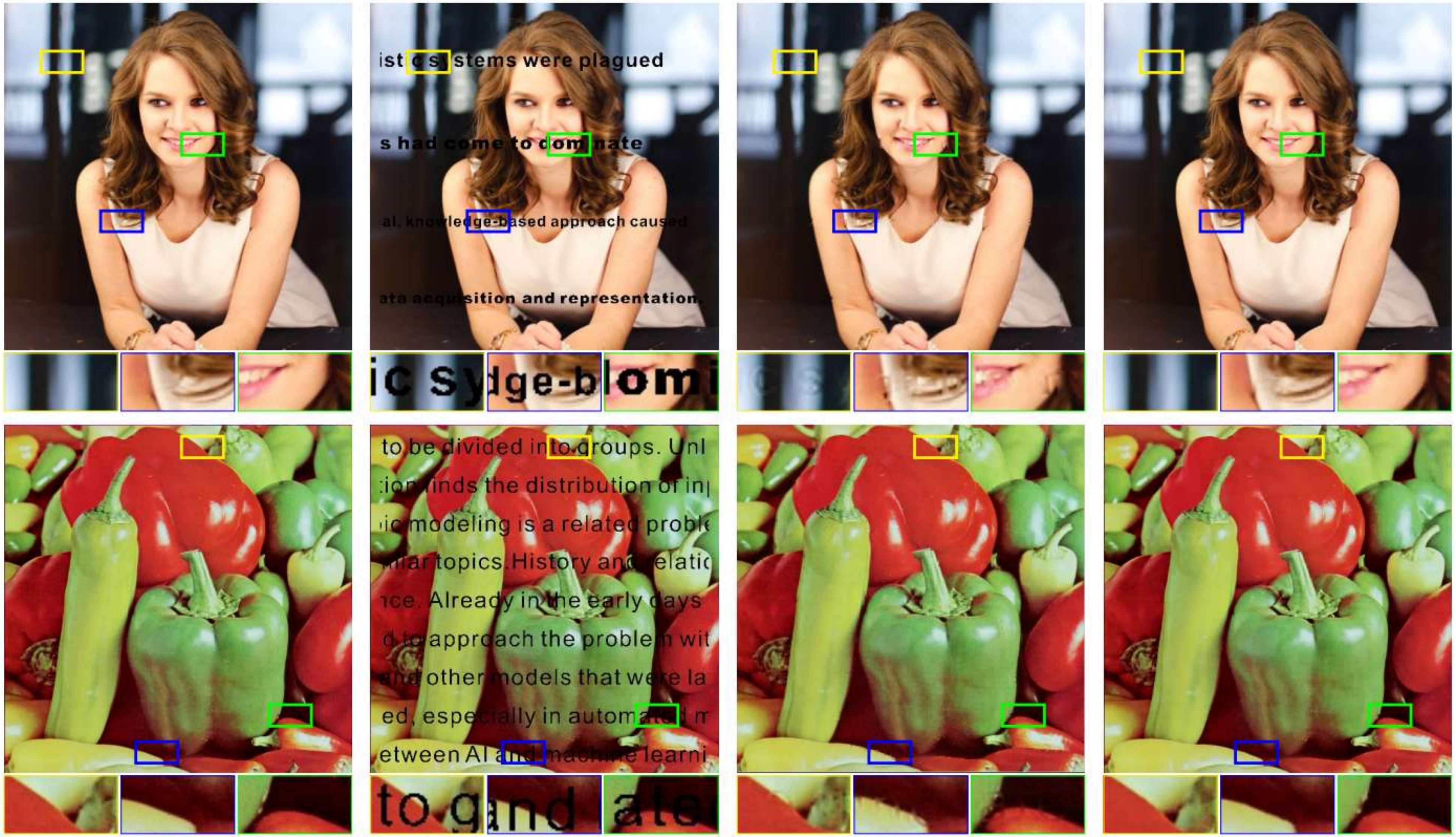
Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

(a) Original image (b) Bicubic, **Not trained** (c) Ours, **Not trained** (d) LapSRN, **Trained** (e) SRResNet, **Trained**

Deep Image Prior

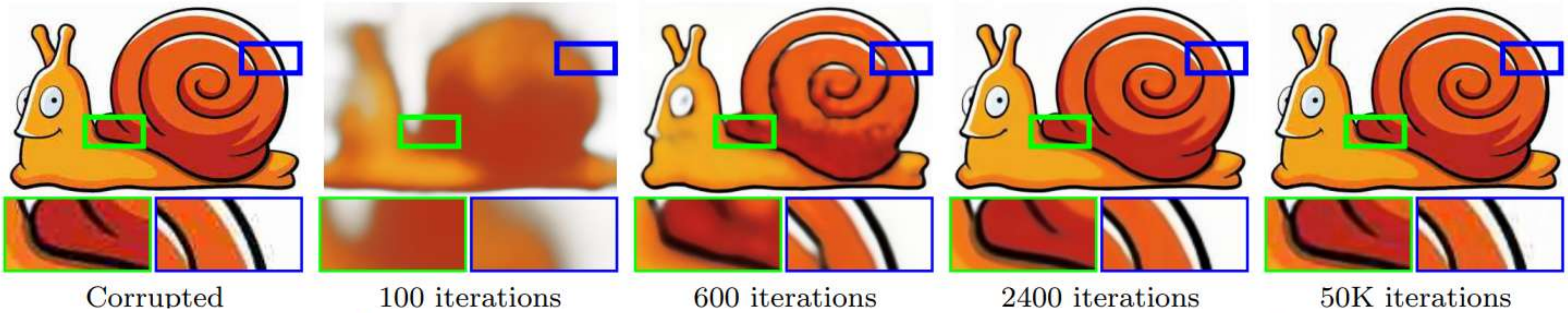
Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

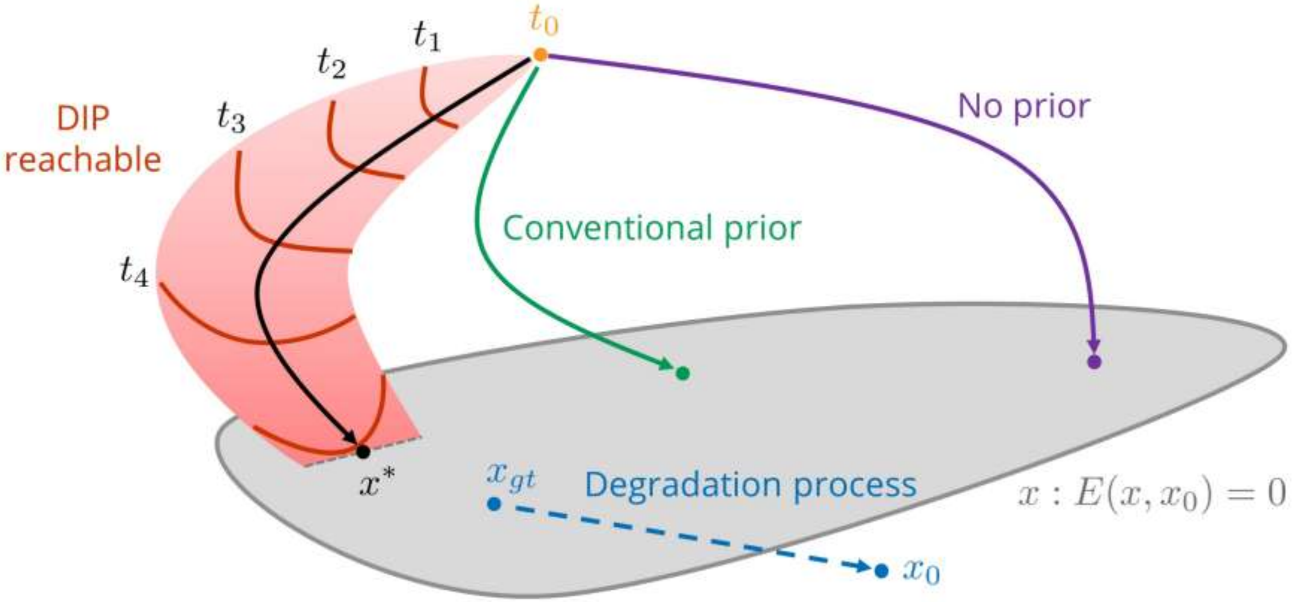


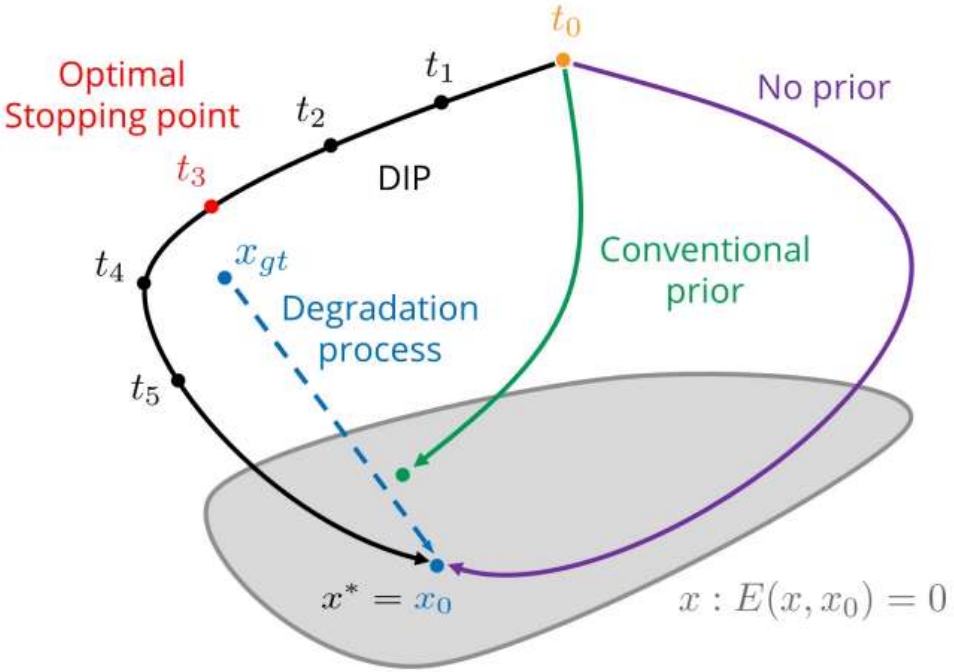
(a) Original image (b) Corrupted image (c) Shepard networks [44] (d) Deep Image Prior

Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

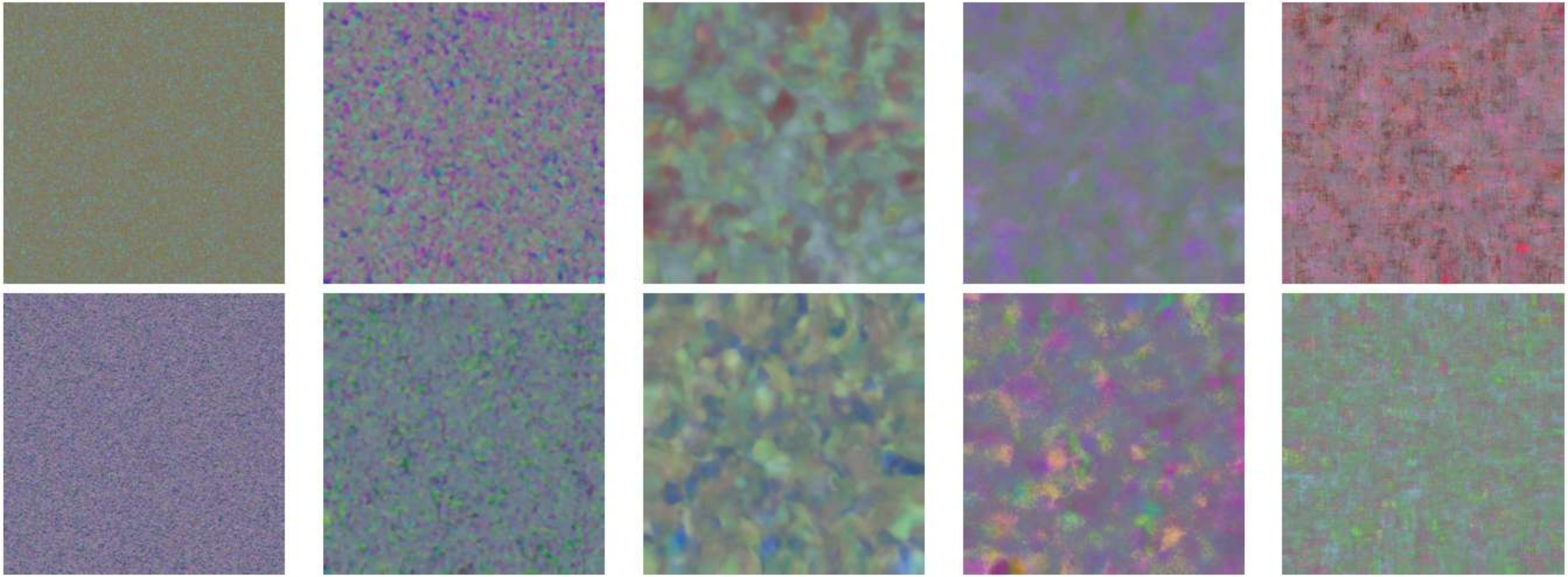






Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.



a) Hourglass-1 b) Hourglass-3 c) Hourglass-5 d) Skip-5 e) Skip-5-nearest

Deep Image Prior

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

(a) Input (white=masked)

(b) Encoder-decoder, depth=6

(c) Encoder-decoder, depth=4

(d) Encoder-decoder, depth=2

(e) ResNet, depth=8

(f) U-net, depth=5

Deep Image Prior

- Simplest/naïve generator, now we have models!
- Do not depend on large scale datasets
- Deep networks have their priors of images

- Slow…
- Still no new content generated

- Related works
  - Tensor Decomposition / Compressed Sensing
  - SinGAN (ICCV 2019 best paper)