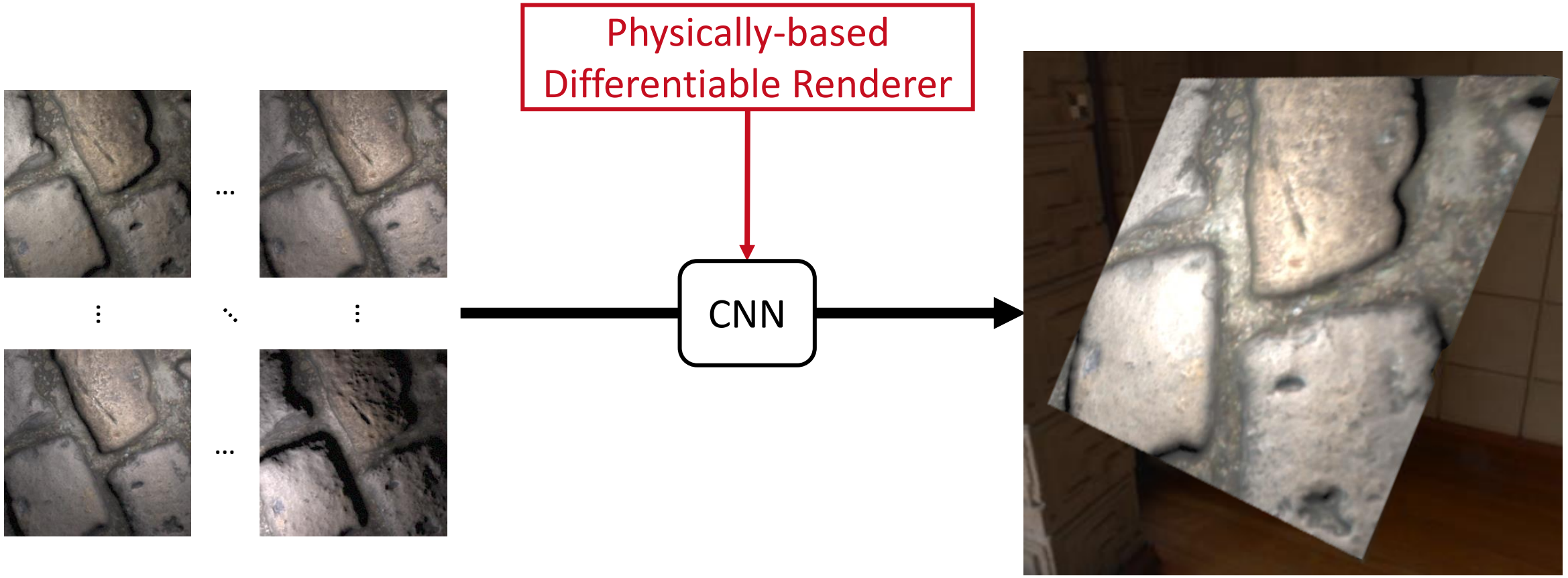
Intermediate Presentation

SVBRDF Estimation using a Physically-based Differentiable Renderer

Markus Andreas Worchel

Recap – Topic



Recap – Schedule

- Download database
- Acquire testing and training code for the network
- Get familiar with papers, source code and data

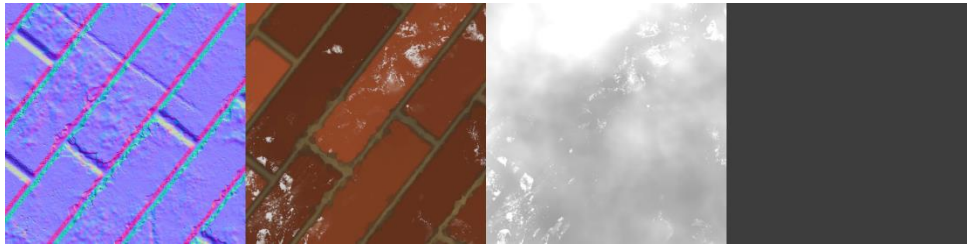
- Acquire code for Mitsuba 2
- Replace rendering layers of the network with Mitsuba 2

- Evaluation (compare with unmodified method)

Data

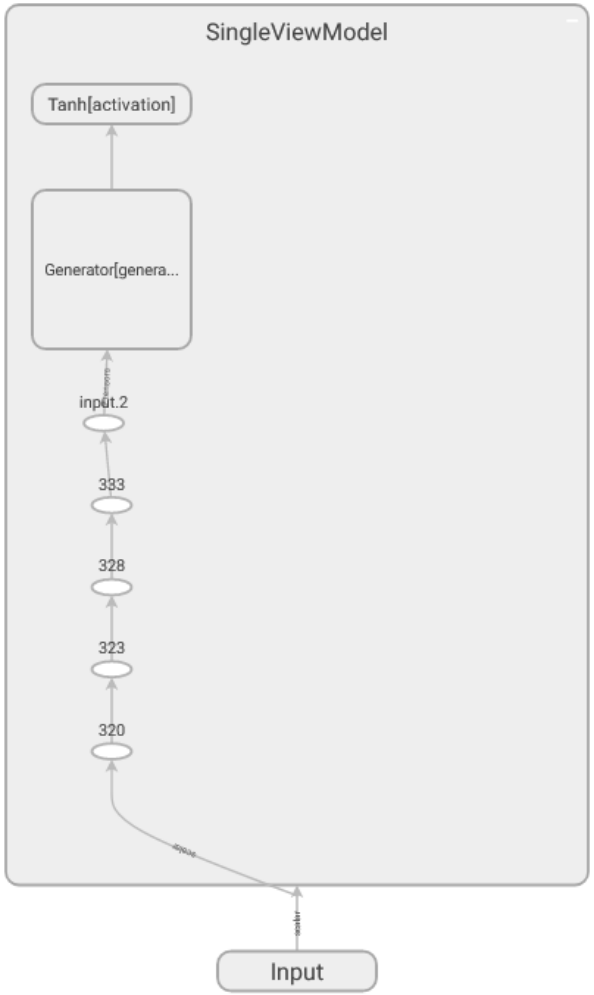
- Single-view database (87 GB)
  - Input image and material variations generated offline
  - Reading implemented ✓

- Multi-view database (1.4 GB)
  - Input images and material variations generated at load time
  - Reading (generation) not yet implemented –

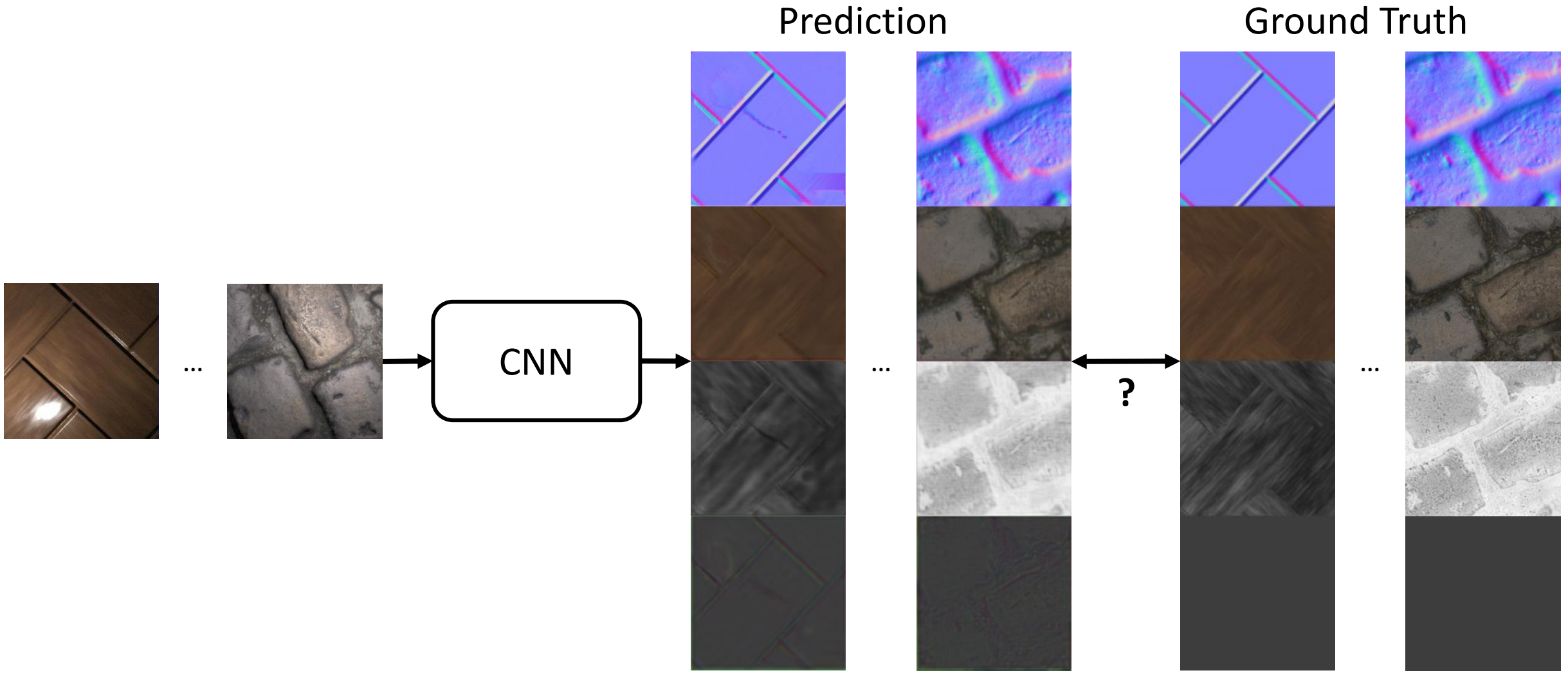


Models

- Single-view model (Deschaintre et al., 2018)
  - Implement in pyTorch ✓
  - Verify model using visualization in TensorBoard ✓
  - Report bugs back to author ✓
- Multi-view model (Deschaintre et al., 2019)
  - Implement in pyTorch ✓
    → Only 40 LoC more than single-view model
  - Verify model using visualization in TensorBoard –

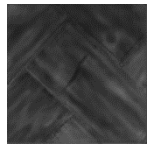


Training Loss

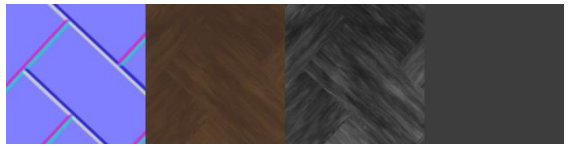
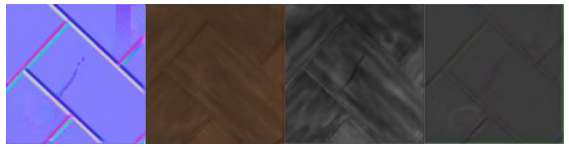


Loss – From L1 to Mixed Loss

- L1 loss between SVBRDF maps was implemented last time

$$L_1 := |\ \text{  }\ - \text{  }\ | + | \text{  } - \text{  } | + | \text{  } - \text{  } | + | \text{  } - \text{  } |$$

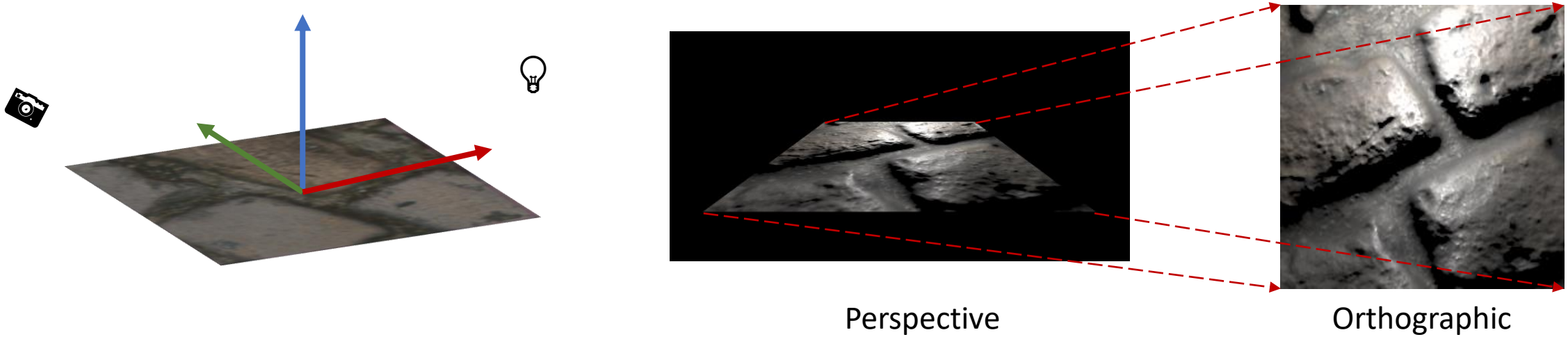
- Now: Mixed Loss

$$L := \underbrace{|R(\ \text{  }\ ) - R(\ \text{  }\ )|}_{\text{rendering loss}} + \lambda L_1$$

- Rendering operator R requires scenes and a differentiable renderer

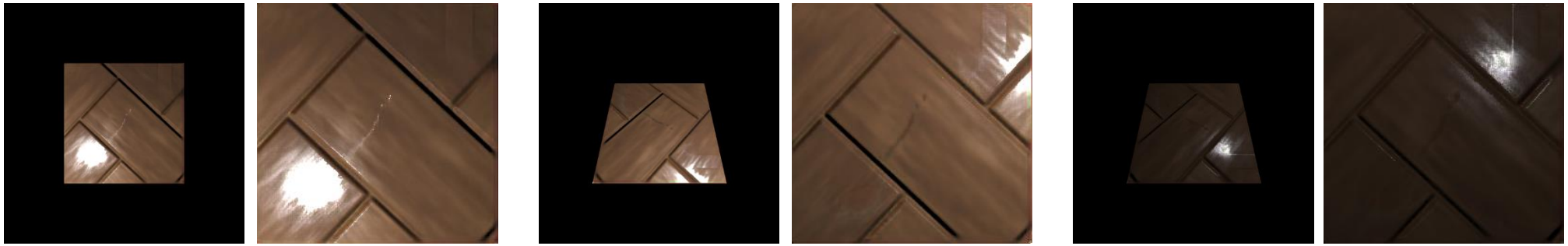
Loss – Renderer

- Implement simple differentiable renderer ✓
  - Only considers direct illumination
  - Lambertian diffuse term
  - Cook-Torrance specular term
  - Renders SVBRDF on flat material patch in the origin (virtual orthographic view)



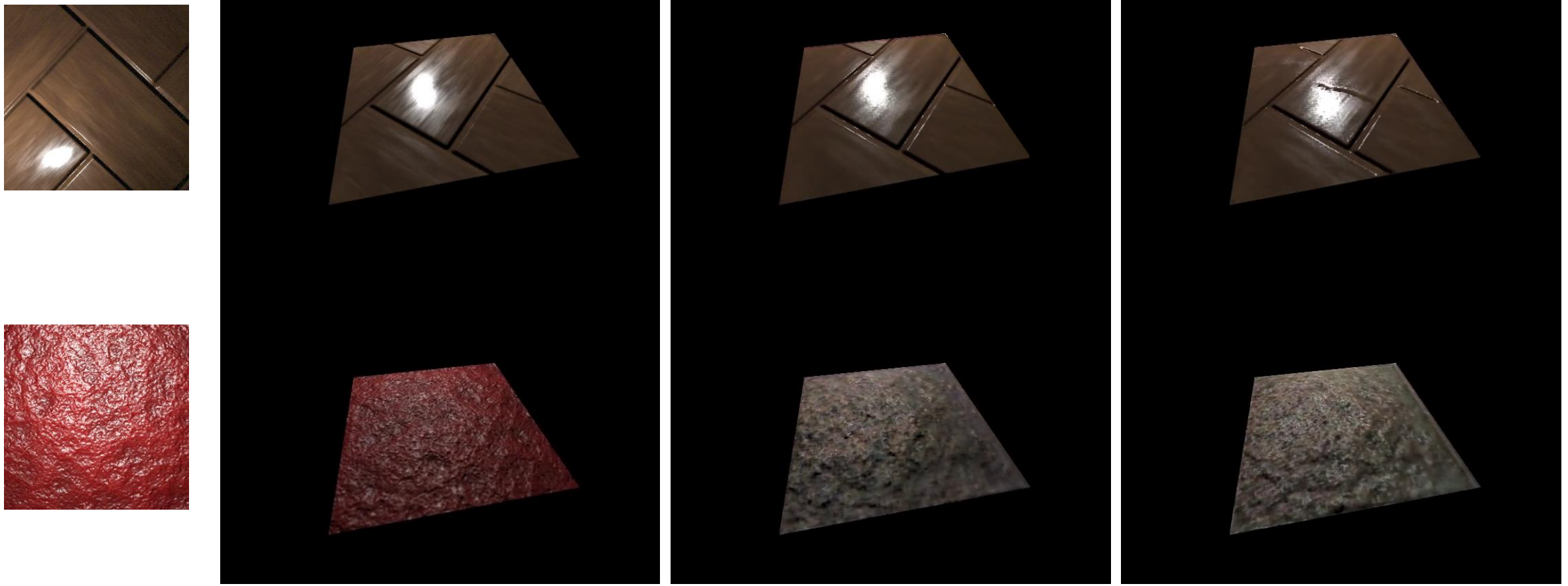
Loss – Scenes

- One camera, one light ✓
- Random camera and light positions per sample in mini batch –
  - Required for unbiased appearance comparison
  - Currently only three fixed configurations:



Loss – Results

Input | Ground Truth | L1 Loss | Mixed Loss (L1 + Rendering)



Replacing the Simple Renderer

- Mitsuba 2 still not released
  - Fall back to Redner ✓
- Redner is not compatible with Windows
  - Patch Redner code (MSVC intrinsics, install script) ✓
  - Send PR to upstream repository ✓
- Uses OpenEXR python bindings (no Windows compatibility)
  - Patch OpenEXRPython code ✓
  - Send PR to upstream repository ✓
- Redner GPU (CUDA) not compatible with Windows
  - Patch Redner Code –
  - Send PR to upstream repository –

Replacing the Simple Renderer – Challenges

- Integration into rendering loss and current structure –
  - Redner scene definition vs. my scene definition
  - Virtual orthographic rendering
- Training time feasibility check –
  - Path tracing is resource and time demanding
  - GPU implementation is probably a must
- Evaluation –
  - Full training of models
  - Qualitative (and quantitative) comparison
  - Training time comparison

Conclusion

- Still some work to do…
  …but most difficult tasks are finished or basis is established

- Redner in action (path tracing our SVBRDFs):