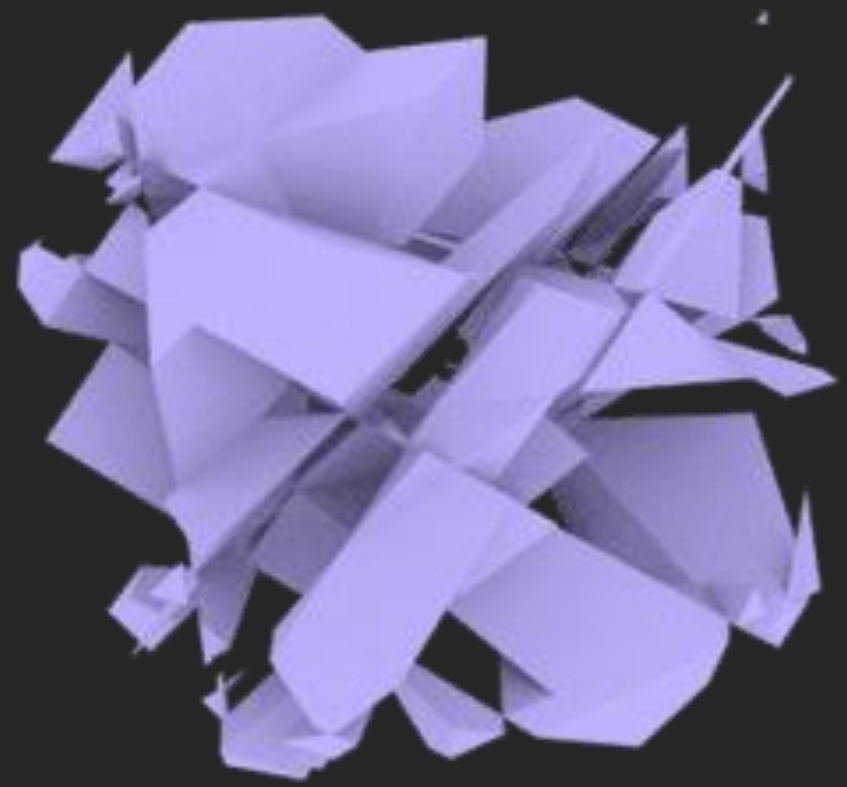
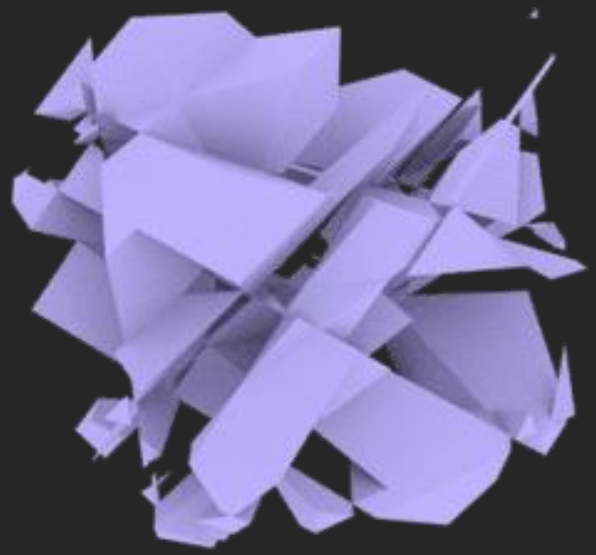
Polyhedral Scene Generator

Tutorial

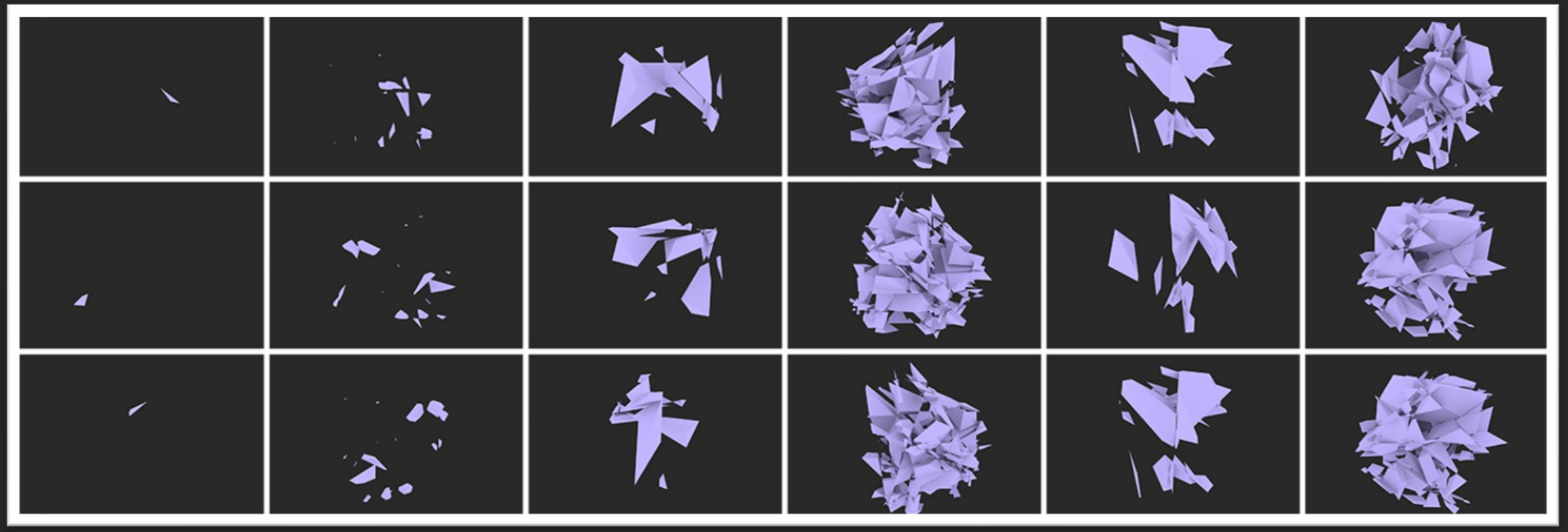


Overview

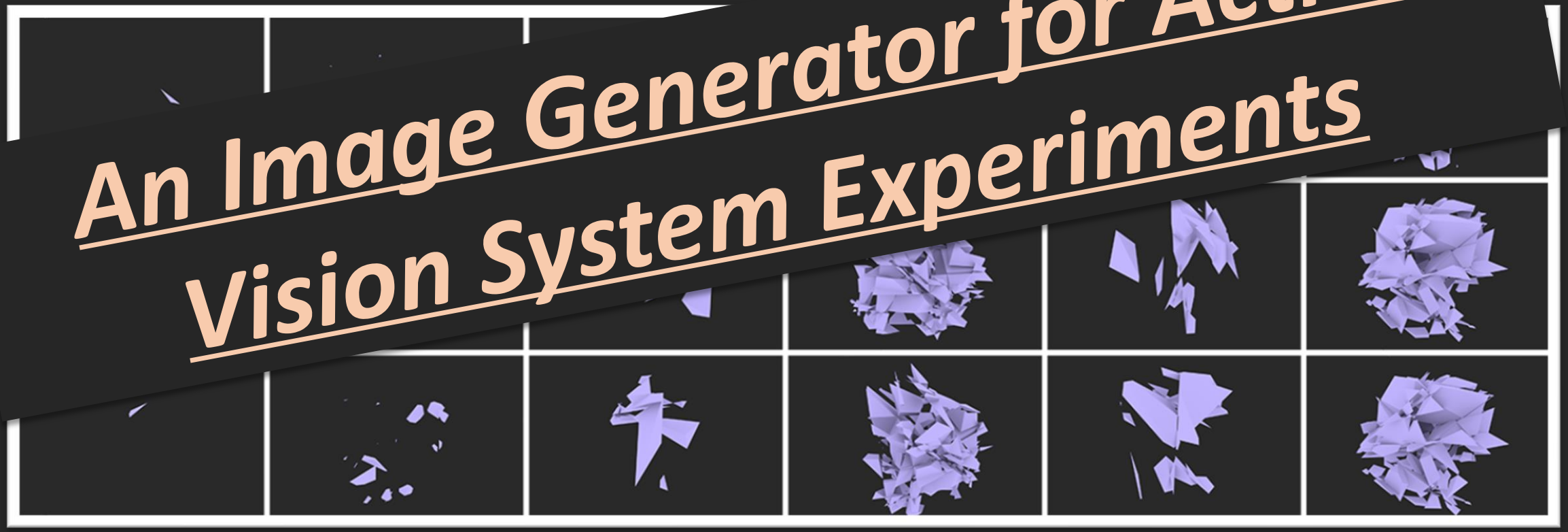
- What is this generator?
- Where can I find more about it?
- Changes we have done since the original paper
- How to use it
- How to get help
- Live Demo



What is it?



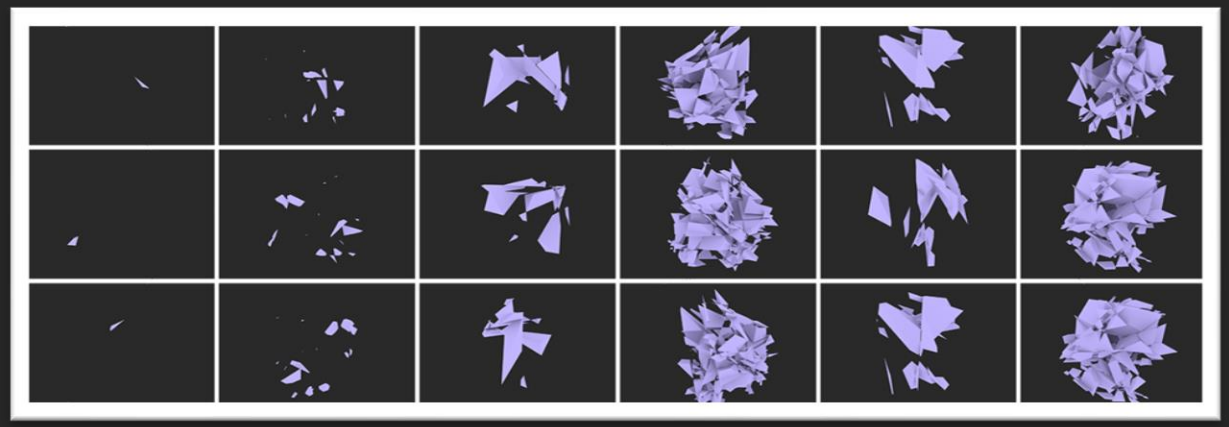
What is it?



An Image Generator for Active Vision System Experiments

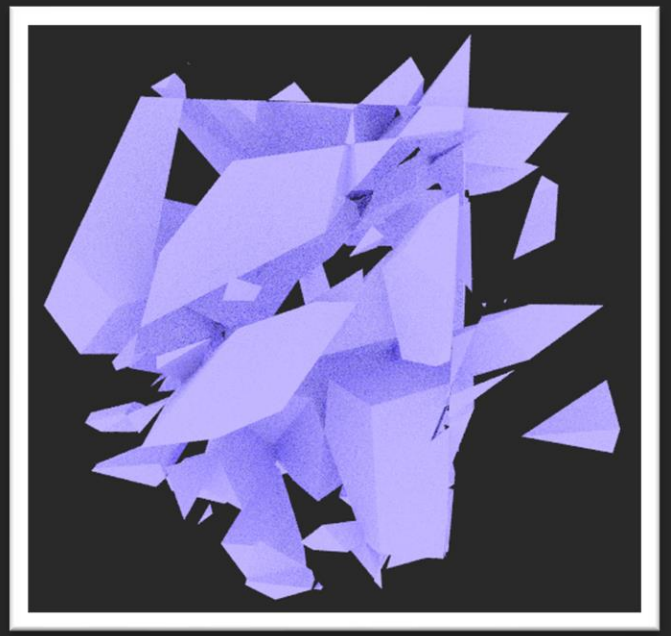
What is it?

- Generates random polyhedral scenes, given:
  - Number of Polyhedra
  - Layout (Separate, Touching, Intersecting)
  - Lighting Condition (Homogeneous & Fixed)
- Provides dataset incl. annotations
- Provides access to generate more observations
  - Allows user to set camera pose
  - Switch between different lighting scenarios
  - Render new observation (webpage & API)



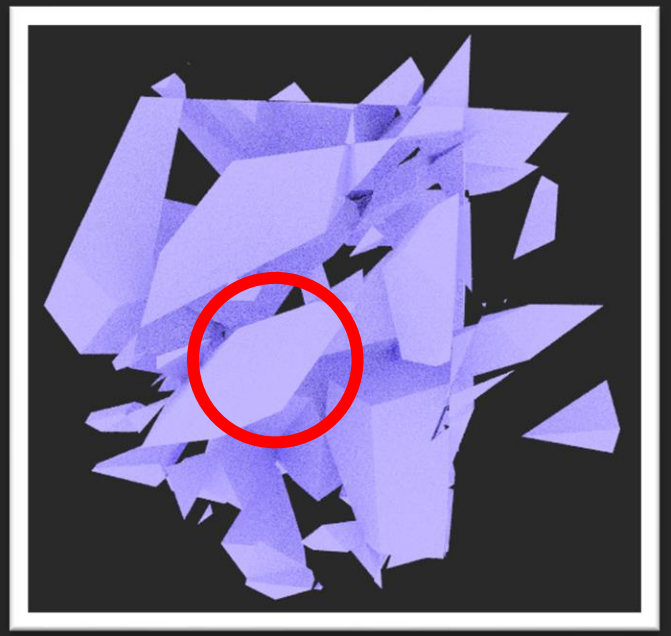
What is it?

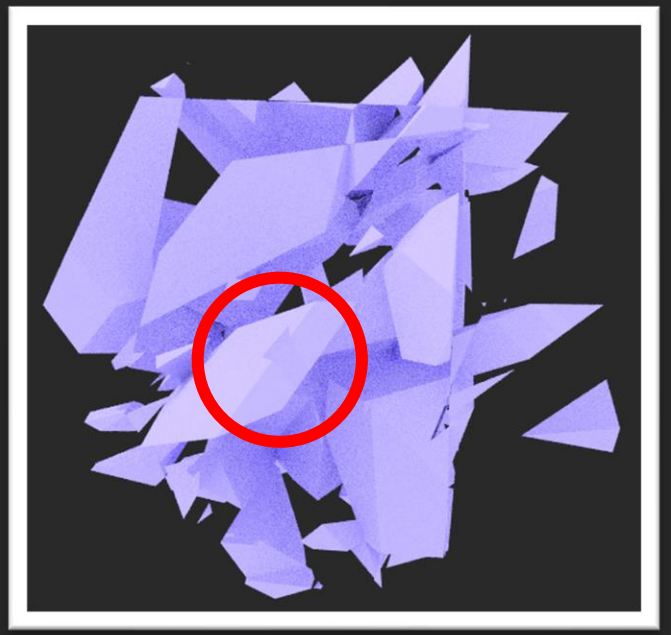
- Generates random polyhedral scenes, given:
  - Number of Polyhedra
  - Layout (Separate, Touching, Intersecting)
  - Lighting Condition (Homogeneous & Fixed)
- Provides dataset incl. annotations
- Provides access to generate more observations
  - Allows user to set camera pose
  - Switch between different lighting scenarios
  - Render new observation (webpage & API)



What is it?

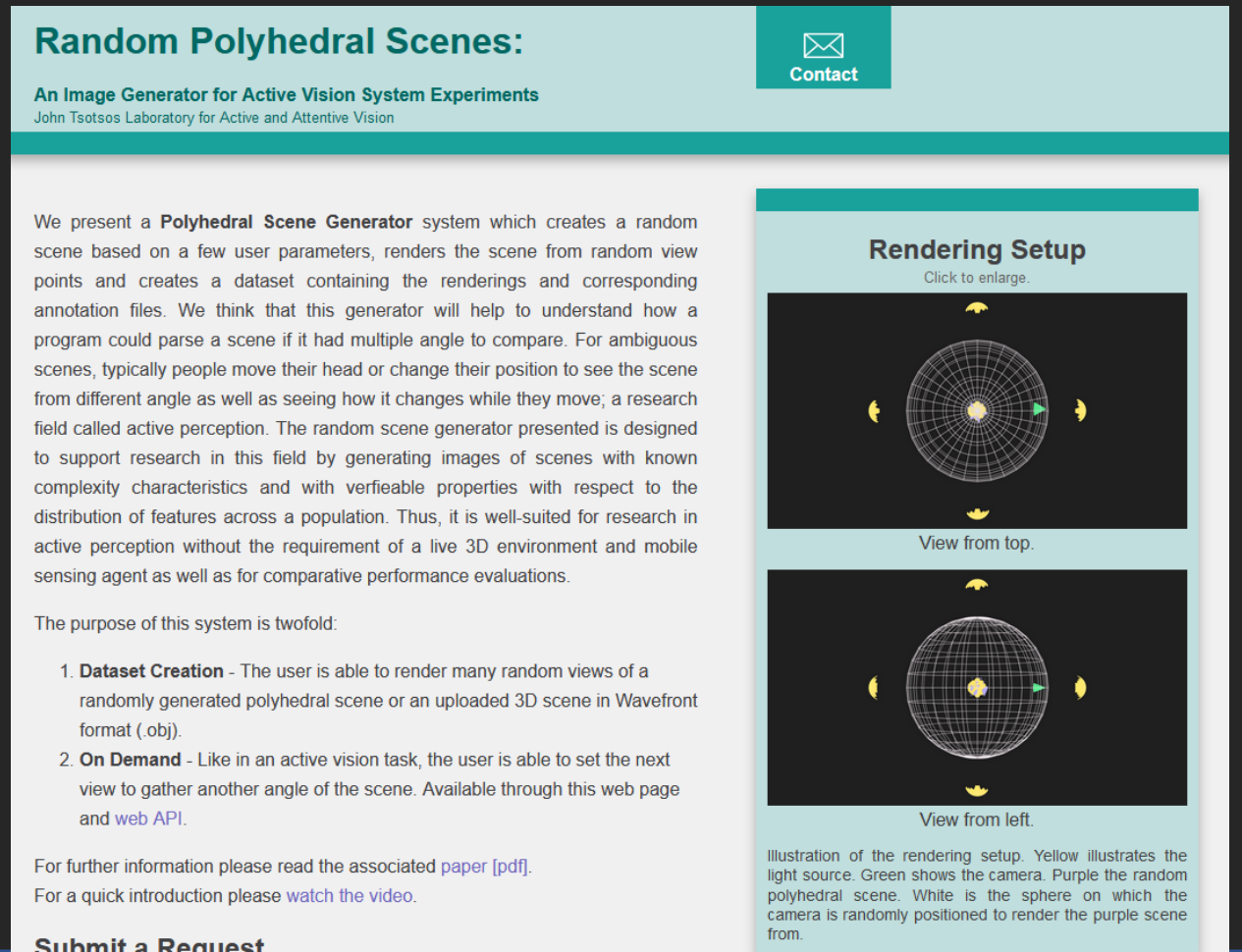
- Generates random polyhedral scenes, given:
  - Number of Polyhedra
  - Layout (Separate, Touching, Intersecting)
  - Lighting Condition (Homogeneous & Fixed)
- Provides dataset incl. annotations
- Provides access to generate more observations
  - Allows user to set camera pose
  - Switch between different lighting scenarios
  - Render new observation (webpage & API)





Where can I find more about it?

- Best starting point is the homepage:
  - https://polyhedral.eecs.yorku.ca/
  - Includes also references to
    - Publication
    - API
    - “walk through” - Video
- Publication:
  - https://arxiv.org/pdf/1803.10100.pdf



Random Polyhedral Scenes:

Contact

An Image Generator for Active Vision System Experiments

John Tsotsos Laboratory for Active and Attentive Vision

We present a **Polyhedral Scene Generator** system which creates a random scene based on a few user parameters, renders the scene from random view points and creates a dataset containing the renderings and corresponding annotation files. We think that this generator will help to understand how a program could parse a scene if it had multiple angle to compare. For ambiguous scenes, typically people move their head or change their position to see the scene from different angle as well as seeing how it changes while they move; a research field called active perception. The random scene generator presented is designed to support research in this field by generating images of scenes with known complexity characteristics and with verfieable properties with respect to the distribution of features across a population. Thus, it is well-suited for research in active perception without the requirement of a live 3D environment and mobile sensing agent as well as for comparative performance evaluations.

The purpose of this system is twofold:

1. **Dataset Creation** - The user is able to render many random views of a randomly generated polyhedral scene or an uploaded 3D scene in Wavefront format (.obj).
2. **On Demand** - Like in an active vision task, the user is able to set the next view to gather another angle of the scene. Available through this web page and web API.

For further information please read the associated paper [pdf].
For a quick introduction please watch the video.

Submit a Request

Rendering Setup

Click to enlarge.

View from top.

View from left.

Illustration of the rendering setup. Yellow illustrates the light source. Green shows the camera. Purple the random polyhedral scene. White is the sphere on which the camera is randomly positioned to render the purple scene from.

Where can I find more about it?

- Best starting point is the homepage:
  - https://polyhedral.eecs.yorku.ca/
  - Includes also references to
    - Publication
    - API
    - “walk through” - Video
- Publication:
  - https://arxiv.org/pdf/1803.10100.pdf

Random Polyhedral Scenes: An Image Generator for Active Vision System Experiments

Markus D. Solbach Stephen Voland
Jeff Edmonds John K. Tsotsos

{solbach, jeff, tsotsos}@eecs.yorku.ca

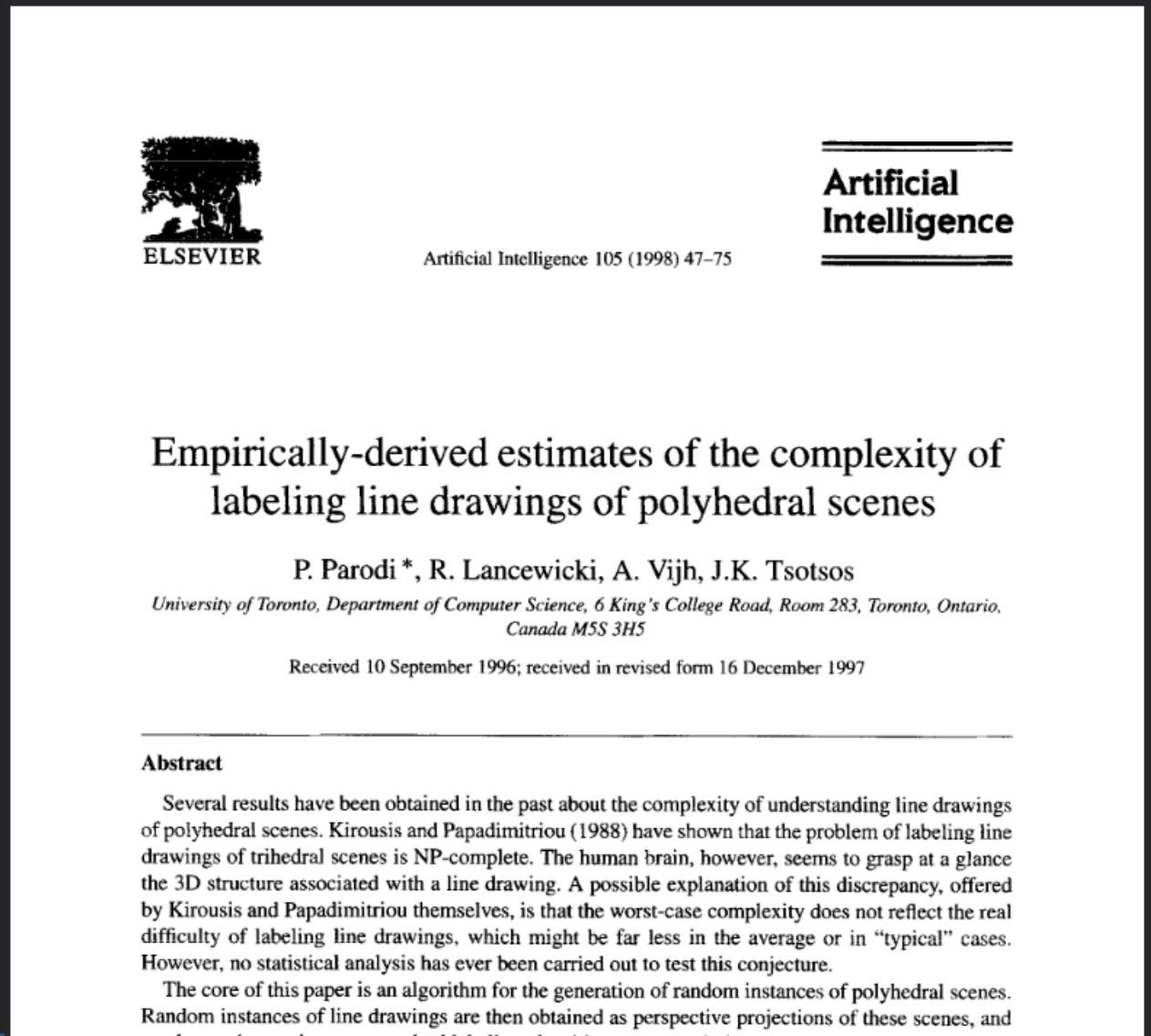
Department of Electrical Engineering and Computer Science
York Univeristy, Canada

Abstract

We present a *Polyhedral Scene Generator* system which creates a random scene based on a few user parameters, renders the scene from random view points and creates a dataset containing the renderings and corresponding annotation files. We hope that this generator will enable research on how a program could parse a scene if it had multiple viewpoints to consider. For ambiguous scenes, typically people move their head or change their position to see the scene from different angles as well as seeing how it changes while they move; this research field is called active perception. The random scene generator presented is designed to support research in this field by generating images of scenes with known complexity characteristics and with verifiable properties with respect to the distribution of features across a population. Thus, it is well-suited for research in active perception without the requirement of a live 3D environment and mobile sensing agent, including comparative performance evaluations. The system is publicly available at https://polyhedral.eecs.yorku.ca.

Where can I find more about it?

- Best starting point is the homepage:
  - https://polyhedral.eecs.yorku.ca/
  - Includes also references to
    - Publication
    - API
    - “walk through” - Video
- Publication:
  - https://arxiv.org/pdf/1803.10100.pdf
- How the scenes are generated:
  - https://tinyurl.com/pyupw8jv



ELSEVIER

Artificial Intelligence 105 (1998) 47–75

Artificial Intelligence

Empirically-derived estimates of the complexity of labeling line drawings of polyhedral scenes

P. Parodi *, R. Lancewicki, A. Vijh, J.K. Tsotsos

University of Toronto, Department of Computer Science, 6 King's College Road, Room 283, Toronto, Ontario, Canada M5S 3H5

Received 10 September 1996; received in revised form 16 December 1997

Abstract

Several results have been obtained in the past about the complexity of understanding line drawings of polyhedral scenes. Kirousis and Papadimitriou (1988) have shown that the problem of labeling line drawings of trihedral scenes is NP-complete. The human brain, however, seems to grasp at a glance the 3D structure associated with a line drawing. A possible explanation of this discrepancy, offered by Kirousis and Papadimitriou themselves, is that the worst-case complexity does not reflect the real difficulty of labeling line drawings, which might be far less in the average or in "typical" cases. However, no statistical analysis has ever been carried out to test this conjecture.

The core of this paper is an algorithm for the generation of random instances of polyhedral scenes. Random instances of line drawings are then obtained as perspective projections of these scenes, and

Changes to the original paper

- “Fixed spotlight” setting (preferred for your assignment)
  - *Before:* Static placement in scene
  - *After:* Moves with the camera. Think of a head-lamp
- Improved Lighting
  - Before: Light might have overexposed certain areas (edges)
  - After: Changes to light intensity and material properties (reflectiveness, …) avoid overexposure.
- Processing time (empty queue)
  - *Before:* 10-30 seconds
  - *After:* 2-5 seconds
- Second Server:
  - nvision3.eecs.yorku.ca:8055/
  - !! Requires VPN to YorkU !!

How to use it

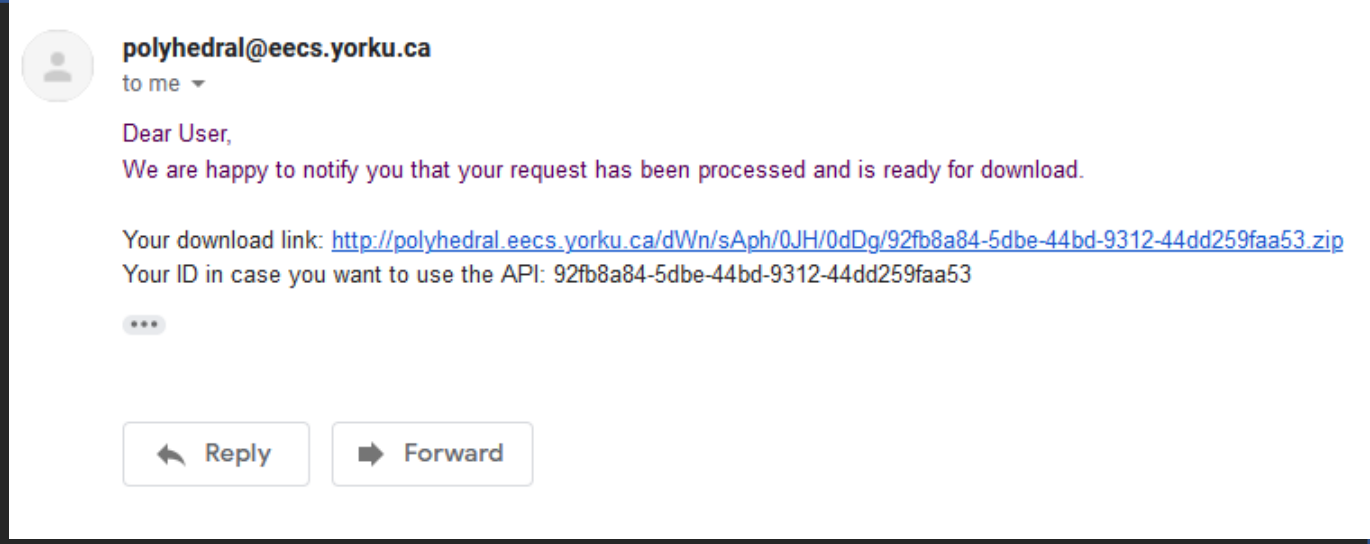
- Choose one of the two Servers
  - https://polyhedral.eecs.yorku.ca/
  - http://nvision3.eecs.yorku.ca:8055
- Scroll down to “Submit a Request”
- Set “Number of Objects” to 1
  - This is what you’ll be dealing with
- Options 2. – 4. don’t matter here
- Enter an email address and click “Submit”
- Get a coffee and check your emails

How to use it

- Choose one of the two Servers
  - https://polyhedral.eecs.yorku.ca/
  - http://nvision3.eecs.yorku.ca:8055
- Scroll down to “Submit a Request”
- Set “Number of Objects” to 1
  - This is what you’ll be dealing with
- Options 2. – 4. don’t matter here
- Enter an email address and click “Submit”
- Get a coffee and check your emails

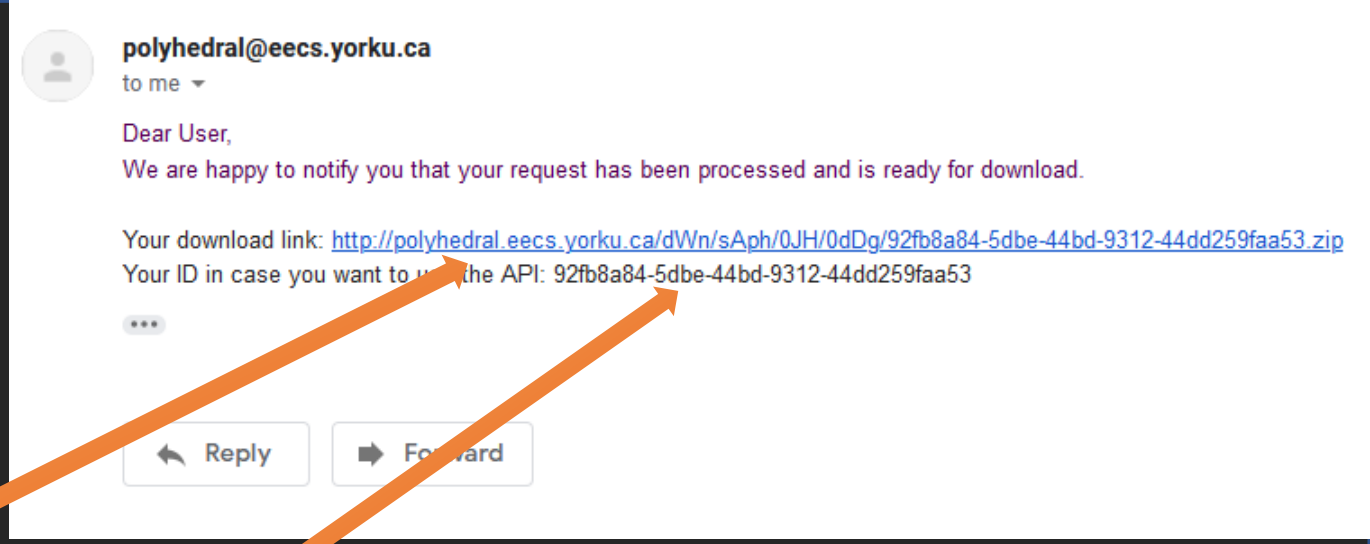
YorkU VPN!

How to use it



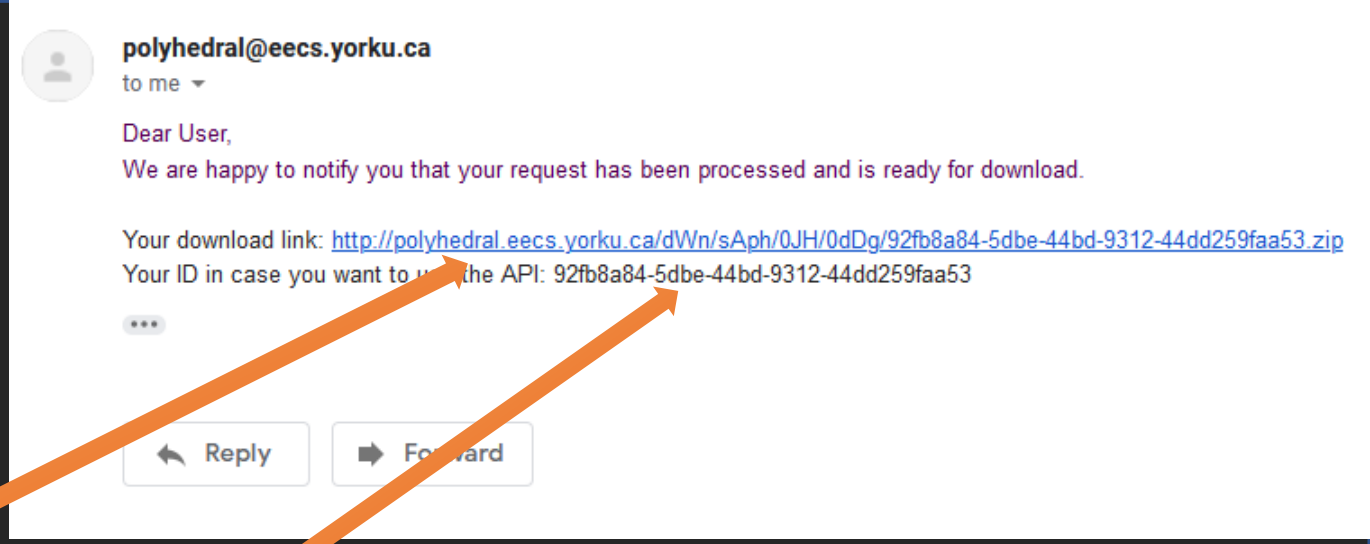
- Check for an email from polyhedral@eecs.yorku.ca
- You will find the download link AND API ID
  - The API ID is necessary to generate further observations

How to use it



- Check for an email from polyhedral@eecs.yorku.ca
- You will find the download link AND API ID
  - The API ID is necessary to generate further observations

How to use it



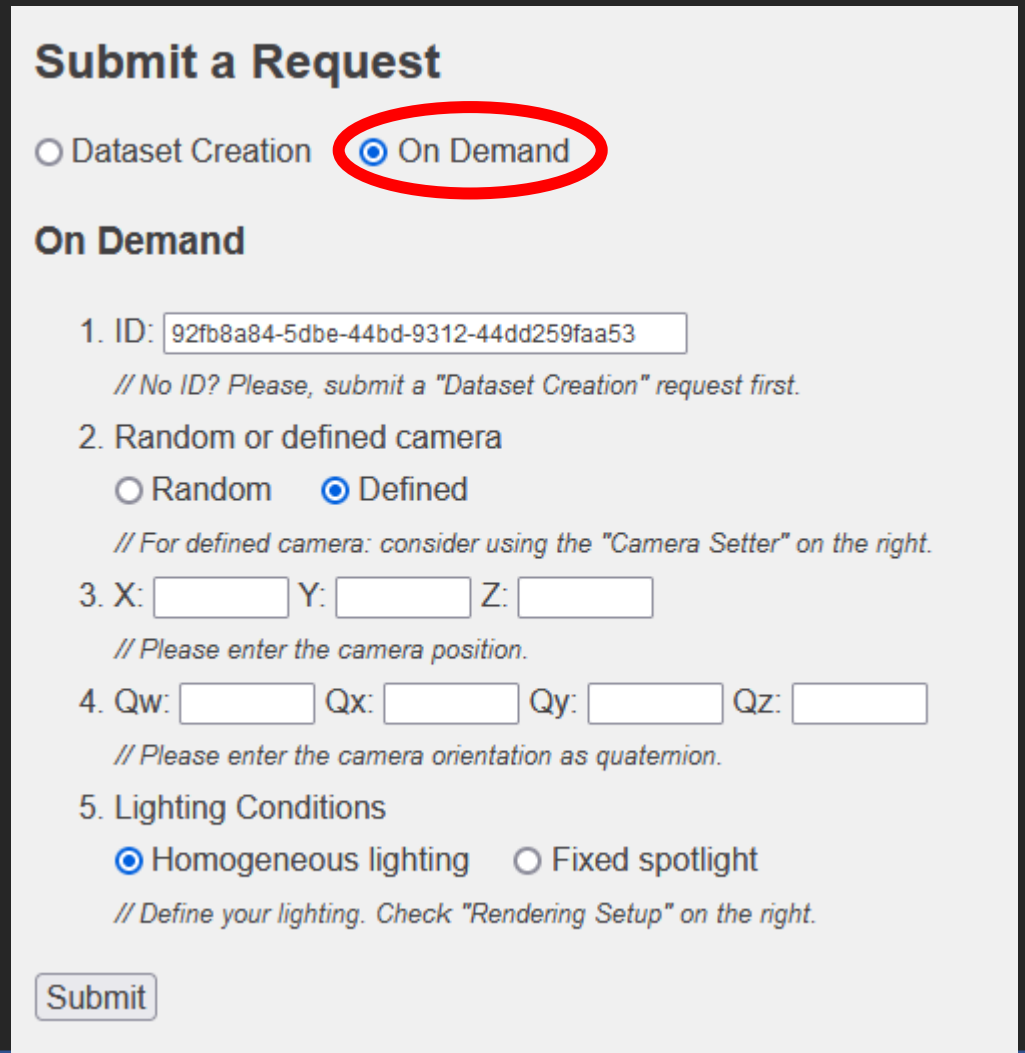
- **Check for an email from polyhedral@eecs.yorku.ca**
- **You will find the download link AND API ID**
  - **The API ID is necessary to generate further observations**
  - **Here:** 92fb8a84-5dbe-44bd-9312-44dd259faa53

How to use it

- To generate additional observations:
  - Webpage
  - API

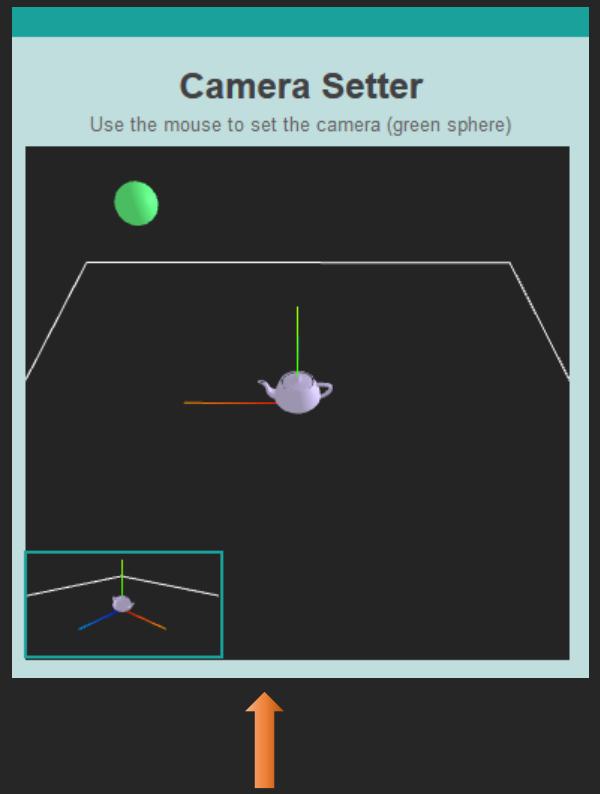
How to use it

- **Webpage:**
  - **Select “On Demand”**
  - **Enter API ID**
  - **Choose the Camera Pose**
    - **Enter numbers or choose *Camera Setter***
  - **Select “Lighting Conditions”**
  - **Click Submit**



How to use it

- **Webpage:**
  - **Select “On Demand”**
  - **Enter API ID**
  - **Choose the Camera Pose**
    - **Enter numbers or choose *Camera Setter***
  - **Select “Lighting Conditions”**
  - **Click Submit**



Move the Sphere to set the camera. Preview shown in bottom left window. It automatically sets the camera pose in the form.

Submit a Request

○ Dataset Creation ◉ On Demand

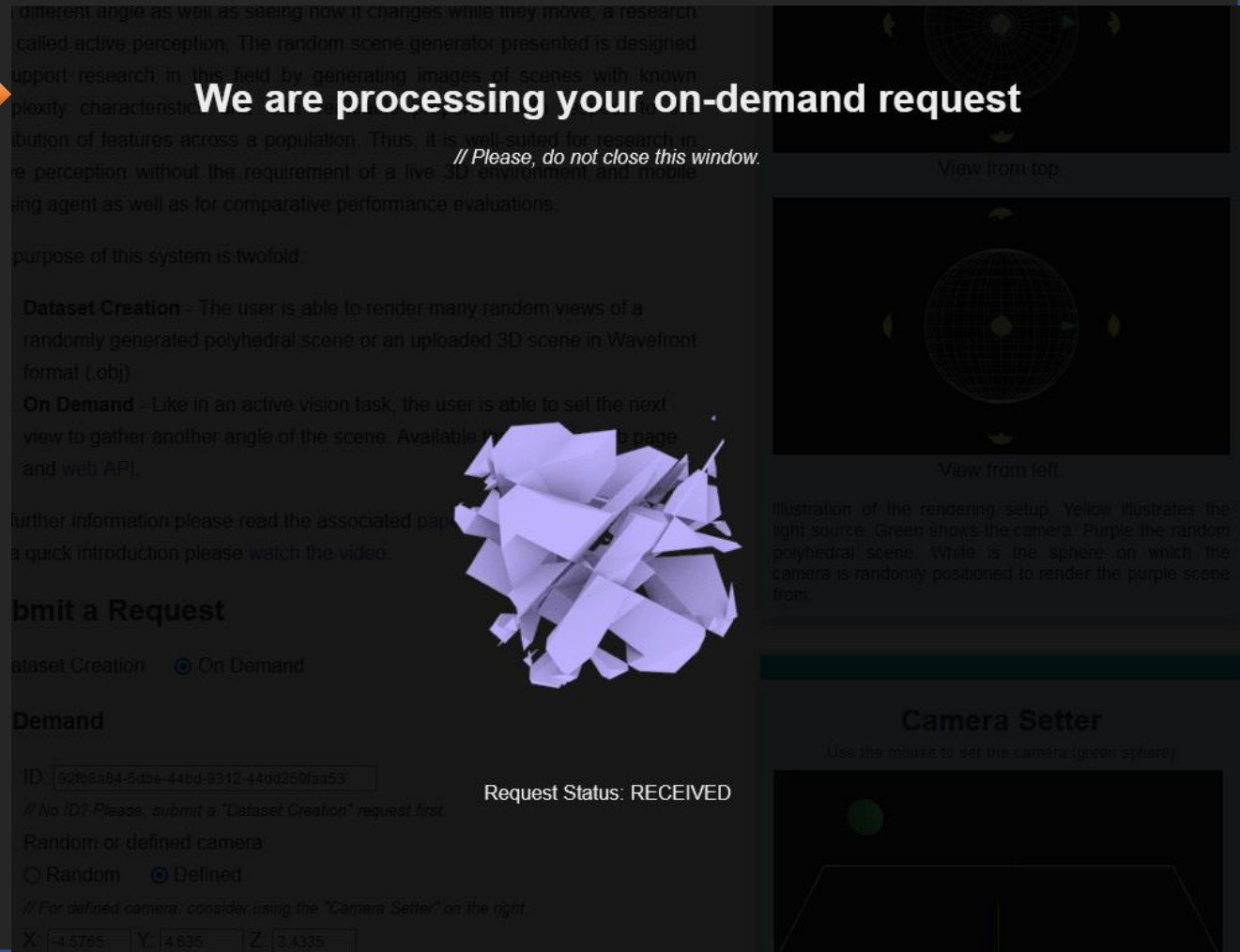
On Demand

1. ID: 92fb8a84-5dbe-44bd-9312-44dd259faa53
   // No ID? Please, submit a "Dataset Creation" request first.
2. Random or defined camera
   ○ Random ◉ Defined
   // For defined camera: consider using the "Camera Setter" on the right.
3. X: Y: Z:
   // Please enter the camera position.
4. Qw: Qx: Qy: Qz:
   // Please enter the camera orientation as quaternion.
5. Lighting Conditions
   ◉ Homogeneous lighting ○ Fixed spotlight
   // Define your lighting. Check "Rendering Setup" on the right.

Submit

How to use it

- The following message will appear
- States:
  - RECEIVED
  - STARTED
  - SUCCESS/FAILED
- A pop-up window will show up with the new observation
  - Make sure to “allow pop-ups”



How to use it

- API:
  - https://solbach.github.io/polyhedral/
  - Examples: Python, MATLAB, Java
  - Full Python example
  - Choose one connection (polyhedral or nvision)
    - nvision will need active YorkU VPN (!)

Choose one!
Note: wss vs. ws

```python
# Author: Markus Solbach (polyhedral@eecs.yorku.ca)
from websocket import create_connection
import io, sys, json, base64
from json import dumps
from PIL import Image
import cv2
import numpy as np

# Create Connection
ws = create_connection("wss://polyhedral.eecs.yorku.ca/api/")
ws = create_connection("ws://nvision3.eecs.yorku.ca:8055/api/") # only available to students at YorkU

# Set Parameters
parameter = {
    'ID':'YOUR ID HERE',
    'light_fixed':'true',
    'random_cam': 'true',
    'cam_x':-0.911,
    'cam_y':1.238,
    'cam_z':-4.1961,
    'cam_qw':-0.0544,
    'cam_qx':-0.307,
    'cam_qy':0.9355,
    'cam_qz':0.16599
}
json_params = dumps(parameter, indent=2)

# Send API request
ws.send(json_params)

# Wait patiently while checking status
while True:
    result = json.loads(ws.recv())
    print("Job Status: {0}".format(result['status']))
    if result['status'] == "SUCCESS":
        break
    elif "FAILURE" in result['status'] or "INVALID" in result['status']:
        sys.exit()

# Processing result
image_base64 = result['image']
image_decoded = base64.b64decode(str(image_base64))

# Create Open CV 2 Image
image = Image.open(io.BytesIO(image_decoded))
cv_image = cv2.cvtColor(np.array(image), cv2.COLOR_BGR2RGB)
cv2.imshow('image',cv_image)
cv2.waitKey(0)
cv2.destroyAllWindows()

# Close Connection
ws.close()
```

How to get help

- If (Question == Technical):
  - solbach@eecs.yorku.ca
- Else:
  - tsotsos@eecs.yorku.ca


Live
Demo!