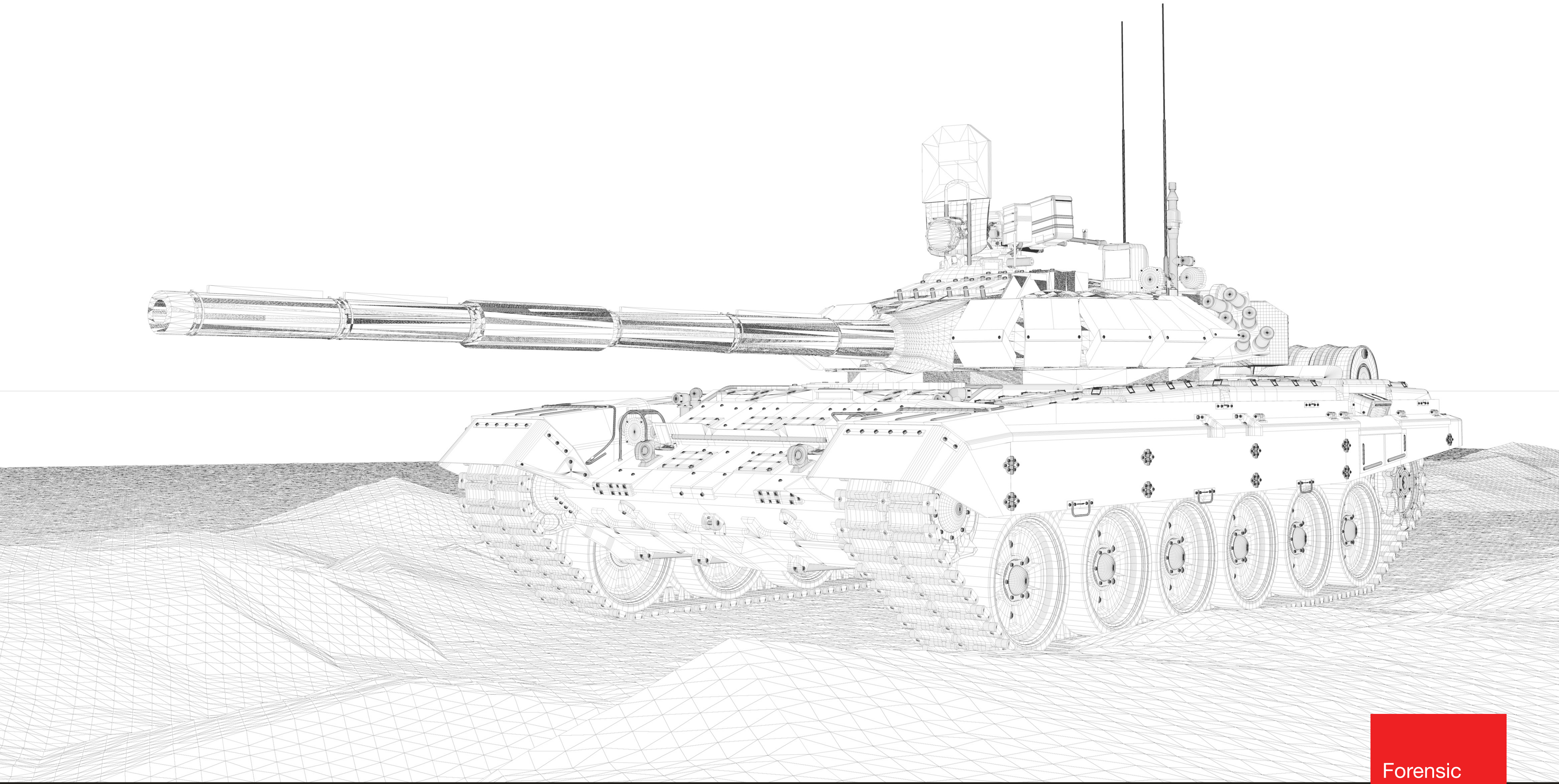
# SYNTHETIC DATA GENERATION

Development of Data Classification Tools



Forensic Architecture

# CONTENTS

# 1.0 INTRODUCTION

**Forensic Architecture is exploring the applications of machine learning in data research. Specifically how machine learning classifiers have effective use in our investigations and can be part of our open-source research framework. We are currently looking at how synthetic data generation might enhance the efficacy of these classifiers.**

## Synthetic Data

Synthetic data generation refers to the ability to generate more data than what may be available for training a classifier.

For instance, if there is a lack of images of a particular object of interest, this object can be digitally modelled and 'photographed' in a parametrised, simulated environment. This can allow for infinite variations of output data that can then be used to train a classifier to detect the object in 'real-world' images.

The topic of Synthetic Data has been of increasing interest in Machine Learning communities, with a plethora of work such as "Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization", 2018 (NVDIA and the University of Toronto). See p.28 for more resources and readings.

## Modelling Software

We have chosen to run our tests in MAXON's Cinema 4D. This 3D modelling and animation software gives us control over animation parameters through its node-based editor XPresso. The node-based approach to parametrising our various variables can be replicated in other software with similar pipelines (Blender, Unity etc.).

Our aim is to develop a modular file set-up in which 3D modelled objects of interest can be inserted at an origin point and images of them rendered.

## Scope

What follows is a guide to our investigation, tests and conclusions. This will be predominantly shown through the files we have set-up in Cinema 4D. We then conduct assessments of image outputs from native renderers (more tests could be done through various other plug-in renderers). The images are then fed to a pre-trained classifier and the classifier's score is subsequently documented.

The assumption is that the reader has experience with 3D modelling software of the kind and an understanding of node-based components, since instructions on software use are not within the scope of this document.

To summarise, we focus on how a simulated environment can be set-up in a 3D modelling software to generate data in as automated and varied a manner possible for the successful reading by (and potential training of) a classifier. Our tests use a case-study investigation where we are looking at identifying specific military hardware in war-zones.

For our current experiments we look at contemporary tank models, for their distinctive shape and size. We plan to further test this on other military hardware as well as dynamic objects such as simulated smoke plumes from missile strikes.

# 2.0 CINEMA 4D FILE OVERVIEW

The file is a product we are continually refining and streamlining. The premise for its structure was to identify a reduced set of variables to then parametrise.

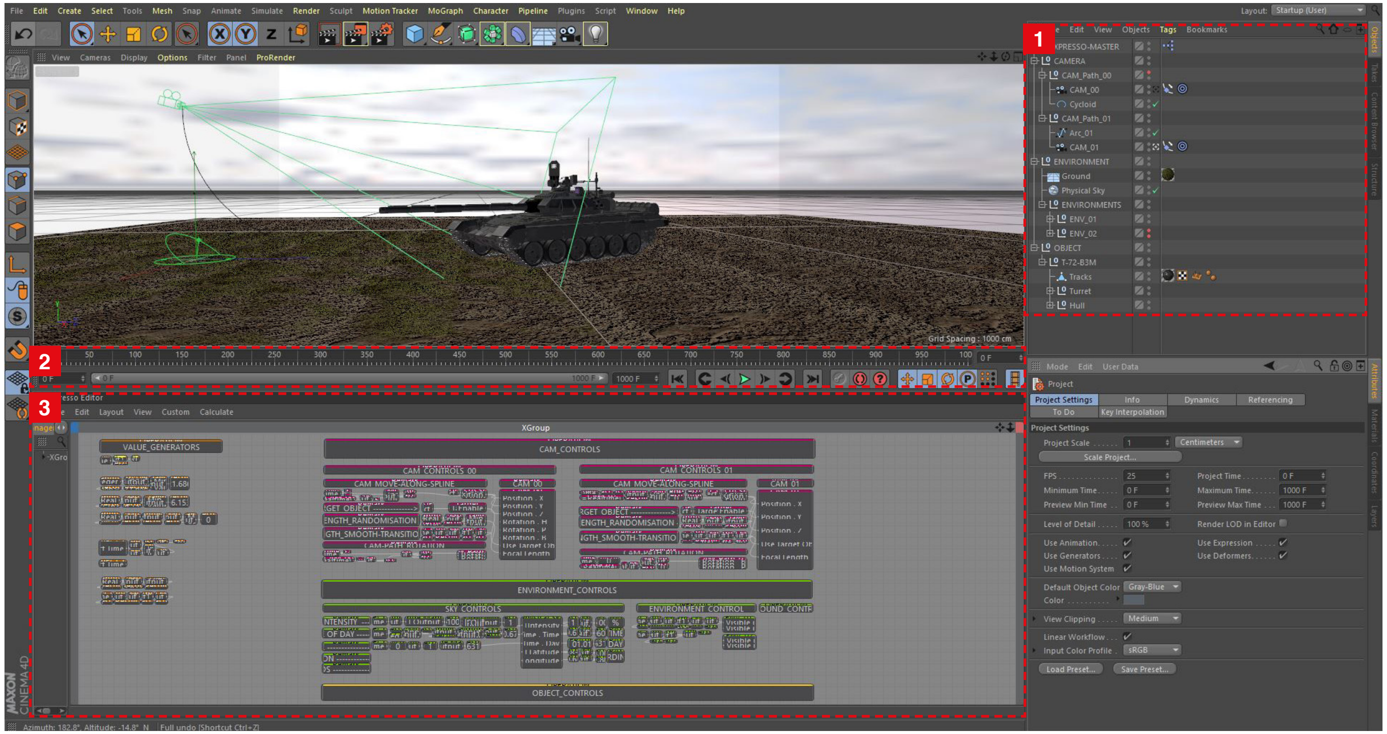
**1. Variable Categories:** CAMERA, ENVIRONMENT and OBJECT form the basis of these variables. These should be treated as fixed layers with more detailed sub-layers.

**2. Varying Across Time:** We decided that the most effective generation of synthetic data would be one that allows the least manual input possible. In other words, when a single render batch can provide enough variation automatically. Therefore our 3D object and its environment should be dynamically simulated to vary across time. This made us decide:

- To use few moving cameras over a number of frames, rather than multiple static cameras in fixed positions.
- To vary the simulation across 1000 frames (a value that can be expanded/contracted). Ideally a min. of 5000 is necessary for training classifiers.

**3. Variable Parameters:** The XPRESSO-MASTER is where we placed all the node-based controls affecting the other three layers.

With the standardised layer categories it should then be easy dragging new 3D objects into the corresponding layer, or creating new environments and cameras, tailored to the context of a classifier's specific data needs.




The template file we built in Cinema 4D for synthetic data generation

## 2.1 LAYER STRUCTURE

- **XPRESSO-MASTER**

  This is a null object with an XPresso tag. The tag allows access to the XPresso node-editor window where we placed all the components controlling the variables across the simulation. (see chapter 3.0 for more)

- **CAMERA**

  Here the global coordinates of any sub-layered cameras can be controlled. All cameras and their camera paths should be contained here.

  - **CAM_PATH_00**
    - CAM_00 — the camera itself with its variables e.g. focal length, position, target etc.
    - PATH_00 — a defined spline that acts as the camera's track for it to move along.
  - **CAM_PATH_01**
    - CAM_01
    - PATH_01

- **ENVIRONMENT**

  Contains anything related to the object's surroundings.

  - **GROUND** — Infinite plane with variable texture
  - **SKY** — C4D Physical Sky with variables e.g. geo-location, sun intensity, time, date etc.
  - **ENVIRONMENTS**
    - ENV_01
    - ENV_02

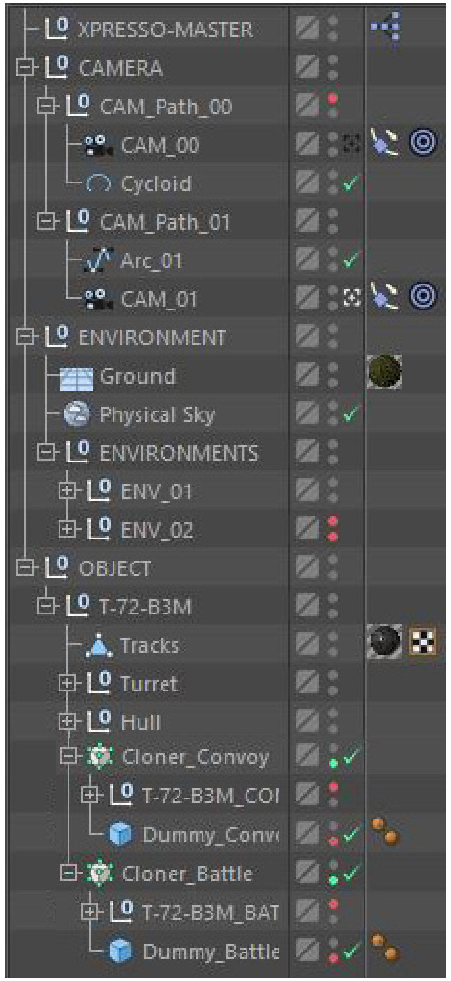
- **OBJECT**

  The global coordinates and other object parameters are controlled from here. Add new 3D objects as sub-layers.

  - OBJ_01
  - OBJ_02
  - **CLONER** — Copies of object scattered and varied

Base Layers


Sub-Layers



### Cameras

Any number of cameras can be set-up. The less cameras the more automated the render process if all output images are a result of a single camera move.

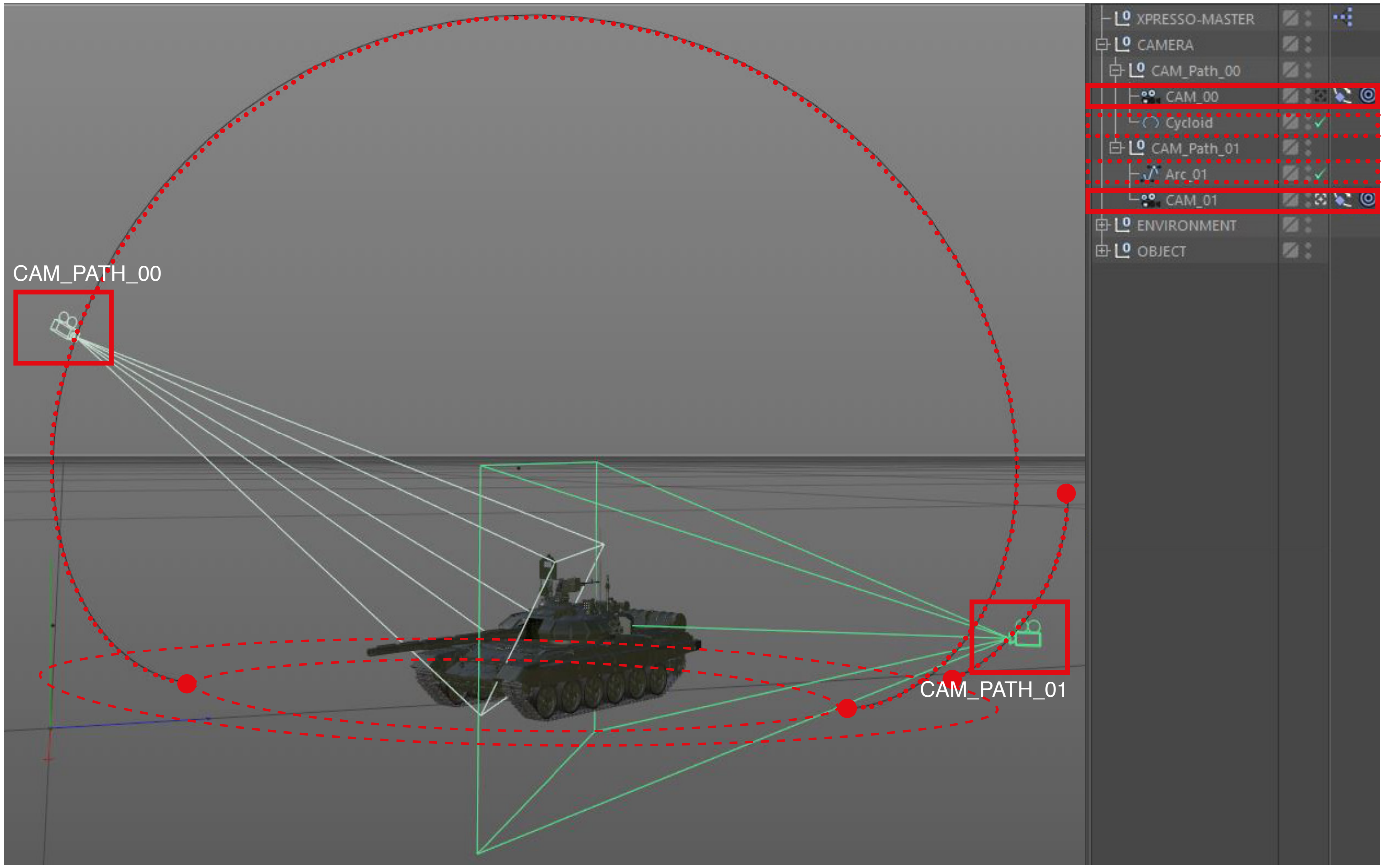
### Tags

The cameras are tagged to 'target' a single object. This means wherever the cameras are tracked to move, the single object will always be in the centre of view in the output images. An 'Align to Spline' tag also sets the camera's path along a prescribed spline.

### Paths

We decided that for our tests with a single target object it makes sense to rotate the camera around the object so as to cover an almost spherical area. This involves having a 'spline' that acts as a path for the camera to move along. Subsequently this 'spline' is also made to rotate 360º laterally around the target object. We later deformed the circular path to vary the camera's distance from the target.

We tested this with CAM_PATH_00. For our use in classifying tanks, camera angles viewing the tank from underneath were not necessary. Hence the half-elliptical arc. In CAM_PATH_01 a more specific refinement accounts for real-world expectation of how a tank would be photographed.




Camera set-up examples

## Ground

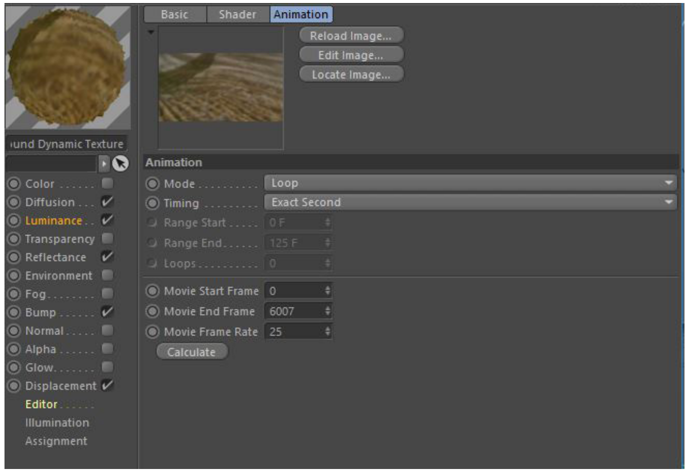
An infinite plane is the base on which other environments can be added on. Its material can be varied. We tried this by applying an animated material texture. This was constructed using a clip showing a time-lapse of a field of wheat that was abstracted in video-editing software (eg: Adobe After Effects) This could be done to abstract other video clips to create animated textures for any object in the scene and change texture over render time.

## Sky

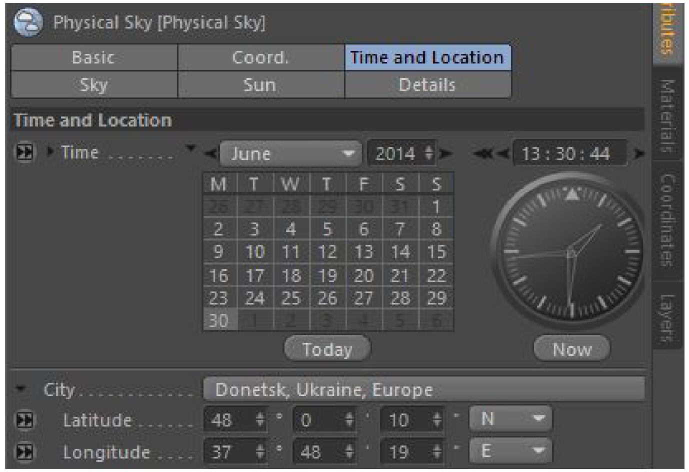
Cinema 4D has an in-built component called 'Physical Sky'. We used this to start testing how to vary sun-paths, locations, time-of-day, clouds etc. across the course of the render time. This was to automate the variation of light and shadow that simulates, as closely a possible, a specific real-world case-study context. For our experiments we used some rural and urban environments. See chapter 3 for more detail on variable parameters.

## Environments

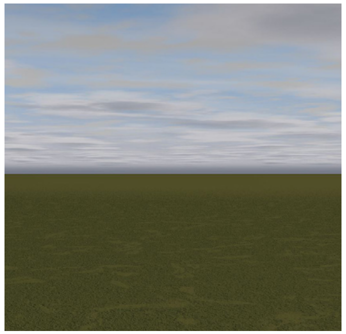
Various environmental tests were specific to the case-study of tanks in rural or destroyed urban setting. The objective is that these can be automatically interchangeable during a single render run to automate the variation.



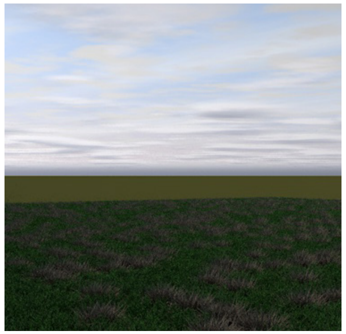

Possibility to use animated textures for abstraction



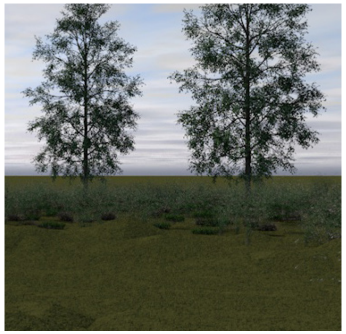

Cinema 4D 'physical sky' parameters



Infinite plane



Dense grassy field



Grassy field and trees

Ruined urban environment

### Object

Objects of interest can form a catalogue of 3D models that can be case-specific. All objects should have a consistent scale and referenced to the origin point in 3D space to make them easily replaceable in the synthetic generation pipeline.

### Animation

Depending on the objects their components could be animated or toggled to vary their appearance across the rendering process. For instance a tank's turret can be parametrised to rotate or some weapons/add-ons can appear/disappear.

The objects could also be scattered across a terrain, randomly varied in position/rotation/clustering, thus adding more variation and accounting for real-world cases not always having the object of interest as the image's central focus.

### Texture

The object's texture can be varied dynamically by applying a video texture UV-mapped to the surfaces.

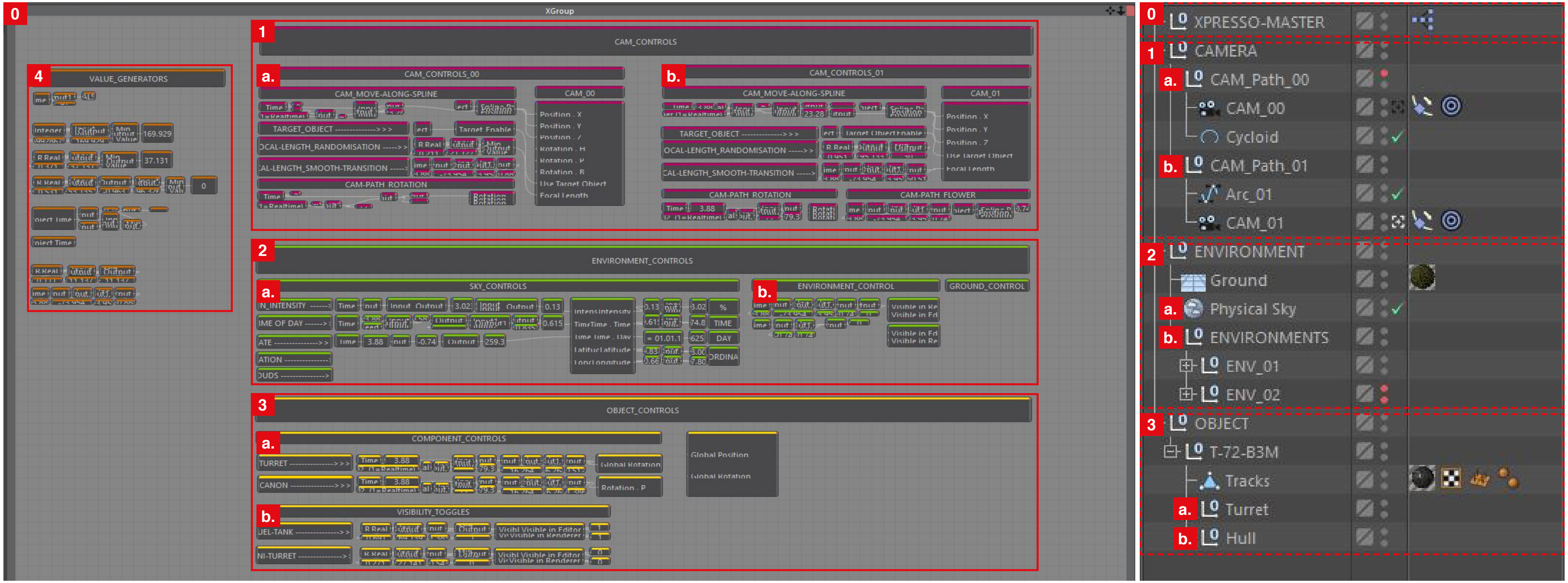
Tank [Type A]

Tank [Type B]

Truck mounted with Missile Launcher [Type A]

Truck mounted with Missile Launcher [Type B]

# 3.0 XPRESSO CONTROLS




**0. XPresso Master**
The main arrangement in the node-editor that includes control components for all variables.

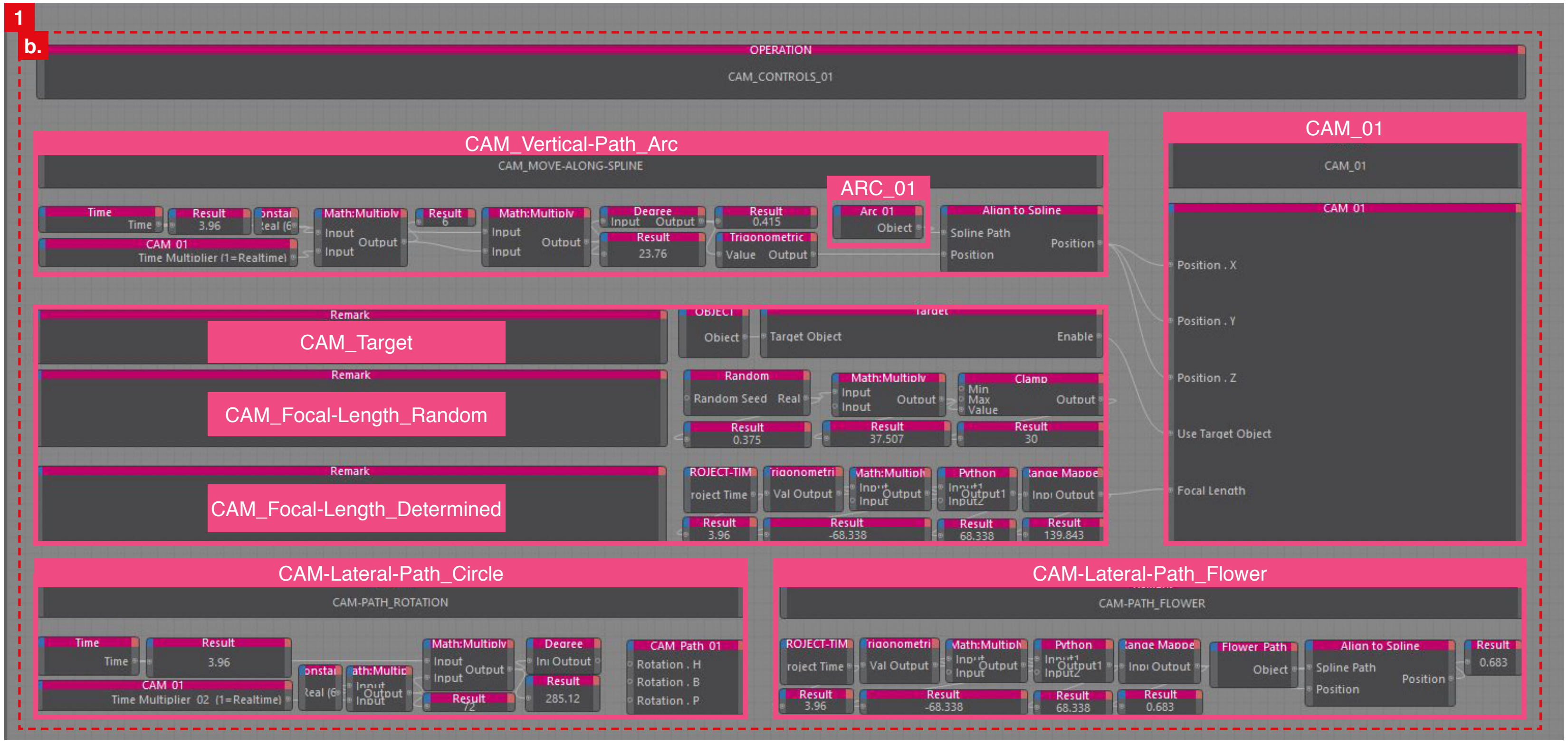
**1. Camera Controls**
Includes all variables related to one or multiple cameras that are set-up.

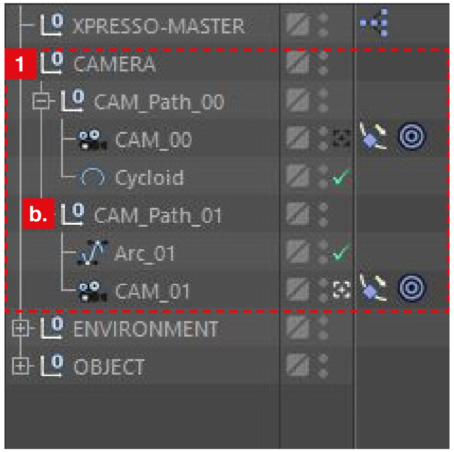
**2. Environments Controls**
Variables related to altering or toggling environments in the scene.

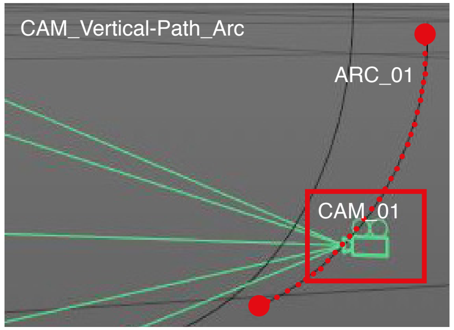
**3. Object Controls**
Variables related to manipulating the object of interest

**4. Value Generators**
Groups of various components that have been set-up to generate different kinds of values (randomly, trigonometricly, etc.) and can be used as inputs to control/manipulate any element that is being varied in our scene.

## 3.1 CAMERA CONTROLS




**CAM_Vertical-Path_Arc:** The node components translate 'Time' input values into co-ordinate positions X-Y-Z along a defined path - in this case ARC_01. A 'Trigonometric' component with a sin curve allows for CAM_01 to fluctuate the co-ordinate positions as time progresses.

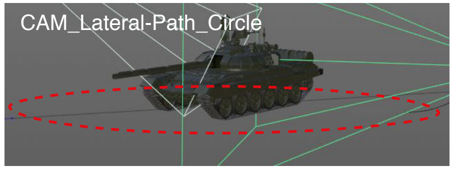
**CAM_Lateral-Path_Circle:** The 'Time' inputs here are globally affecting rotation values H-B-P of the whole layer [b.] CAM_Path_01 and rotate it laterally across time.

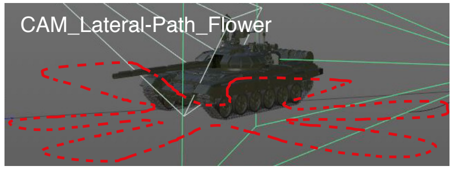
**CAM_Lateral-Path_Flower:** This other lateral-path option can move the whole layer [b.] CAM_Path_01 along a pre-determined spline path - in this case a 'flower' shape path, that is preferred for variation in distance between camera and target object.

**CAM_Target:** Sets the camera to focus on object

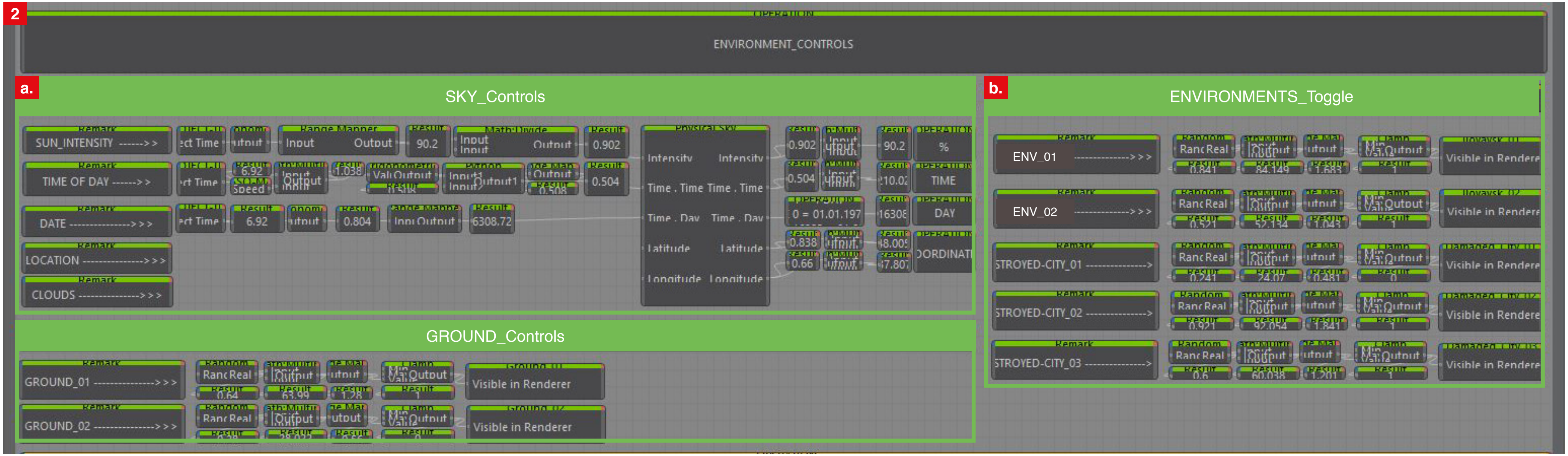
**CAM_Focal-Length_Random:** A randomiser generates values that are re-mapped into a range of likely focal length values.

**CAM_Focal-Length Determined:** No-random values of 'Project Time' are run through a 'Trigonometric' component to cycle back and forth within a re-mapped range of likely focal lengths.



## 3.2 ENVIRONMENT CONTROLS




**Sky_Controls:**

**Sun Intensity:** Uses ‘Project Time’ to change light intensity values of the sun. Values are re-mapped to 0-1 [percentage of intensity].

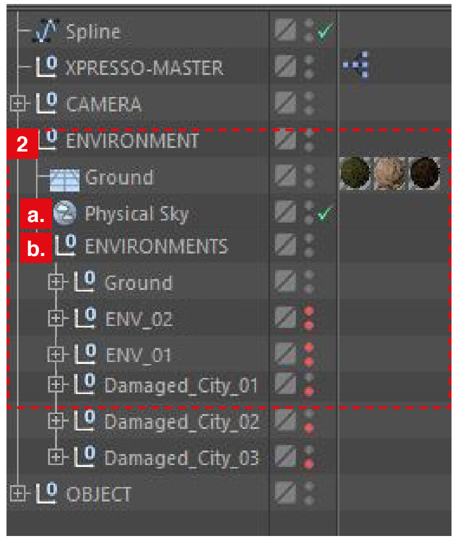
**Time of Day:** Time values were tested to show that 1 hr = 0.042. Midnight 24:00 = 1.000. Using a ‘Trigonometric’ component we alternate time values between 0.417 [10:00] and 0.667 [16:00] since for our specific case-study it was important to detect tanks in day-time conditions.

**Date:** With some input-output tests we determined that 1 day = 1 value. But date inputs [DD.MM.YYYY] correspond to larger output numbers [eg: input date 17.08.2014 gives an output 16299]. Based on this we could determine that output value ‘0’ is equal to input date 01.01.1970. We then re-mapped values from the ‘Project Time’ component to alternate across a range of dates. For example, for dates between 17.08.2014 - 02.09.2014 we needed values between 16299 - 16315.

**Ground_Controls:** Toggles the appearance of different ground conditions randomly across time.

**Environments_Toggle:** This randomly toggles the appearance of different contexts (3D surroundings) across time. For our case-study we had mostly rural fields but included a ruined urban context to vary the backdrop further.

**Note:** It is possible to vary even more environmental parameters such as location, weather (clouds/fog) etc. Controlled testing showing the impact of such variations on classification and training is yet to be done.



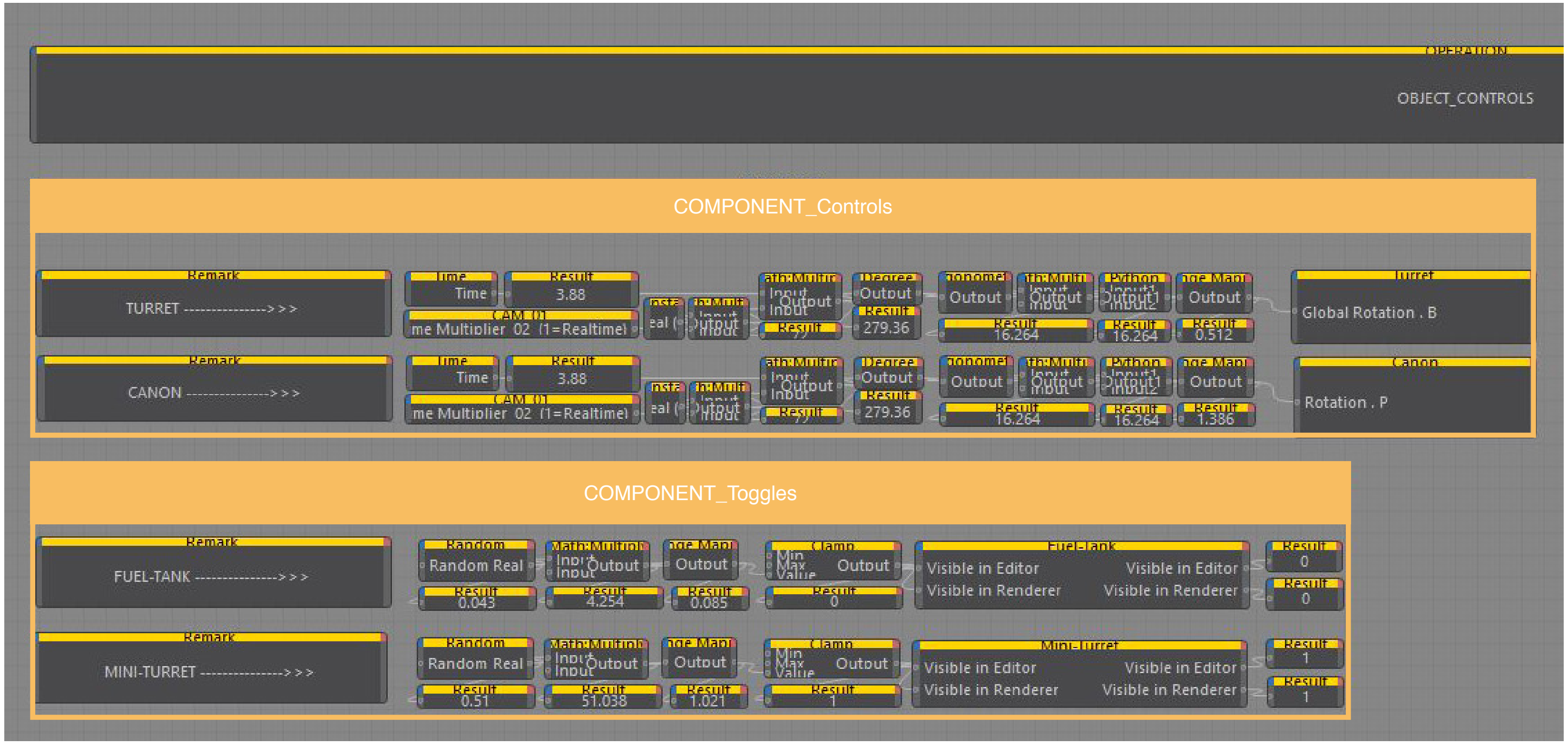
## 3.2 ENVIRONMENT CONTROLS

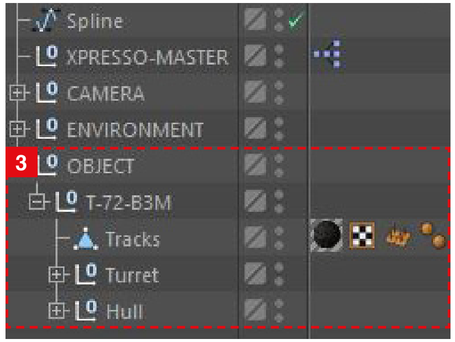
ENV_01 and Damaged-City_01

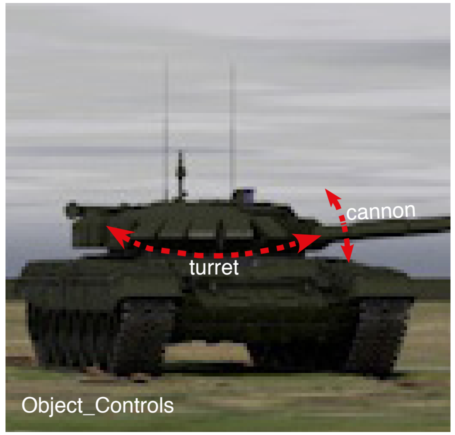
ENV_01 and Damaged-City_02

Damaged-City_02

## 3.3 OBJECT CONTROLS



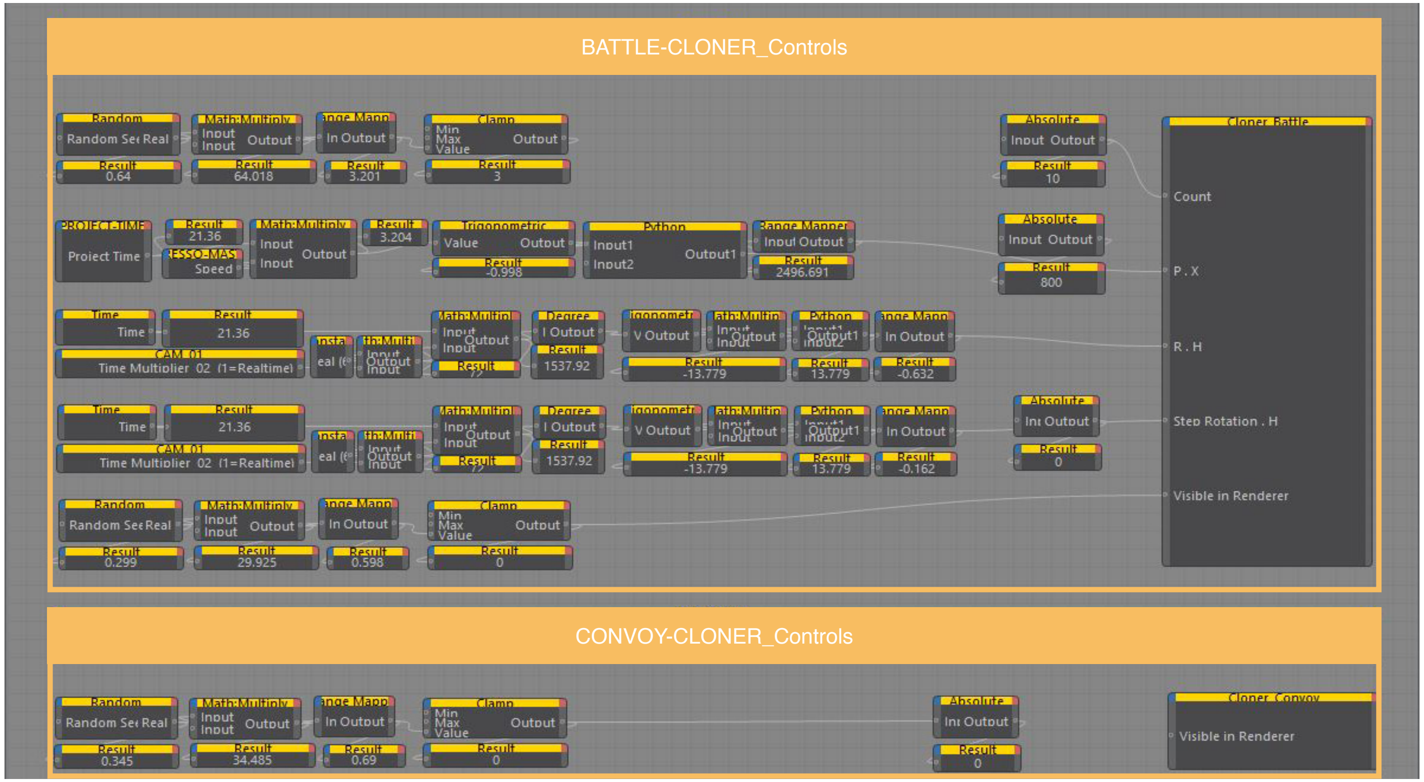



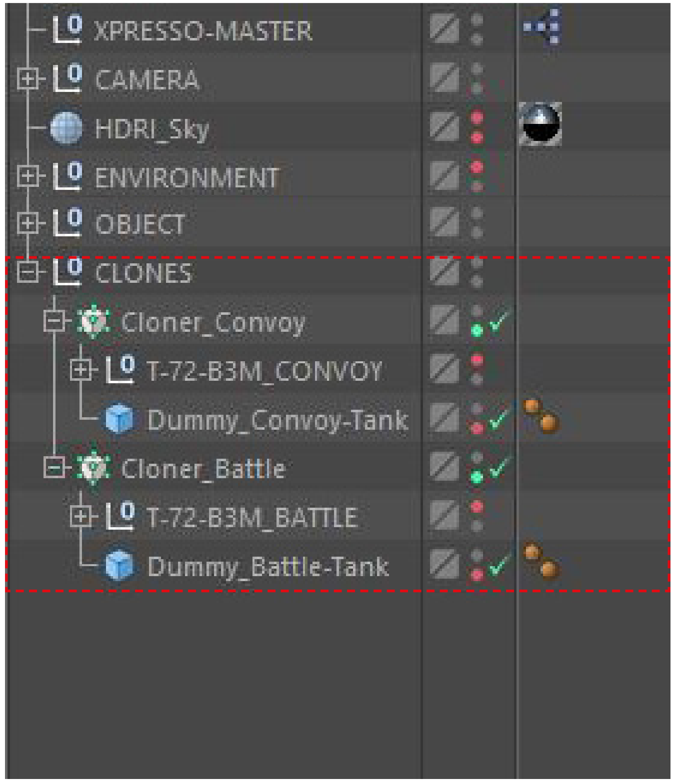




**Component_Controls:** The object of interest may have sub-objects that can be controlled for variation. In our case, the tank's turret can be rotated as well as it's canon over time.

**Component_Toggles:** This allows for certain sub-objects of a larger object to be altered or vanish in some of the generated images. For now a randomiser is used to toggle when certain components of our tank appear or not. A smaller turret mounted onto it, or a fuel-tank, are components that can be toggled to further add to the variability of the object's form in the generated dataset of images.

## 3.3 OBJECT CONTROLS

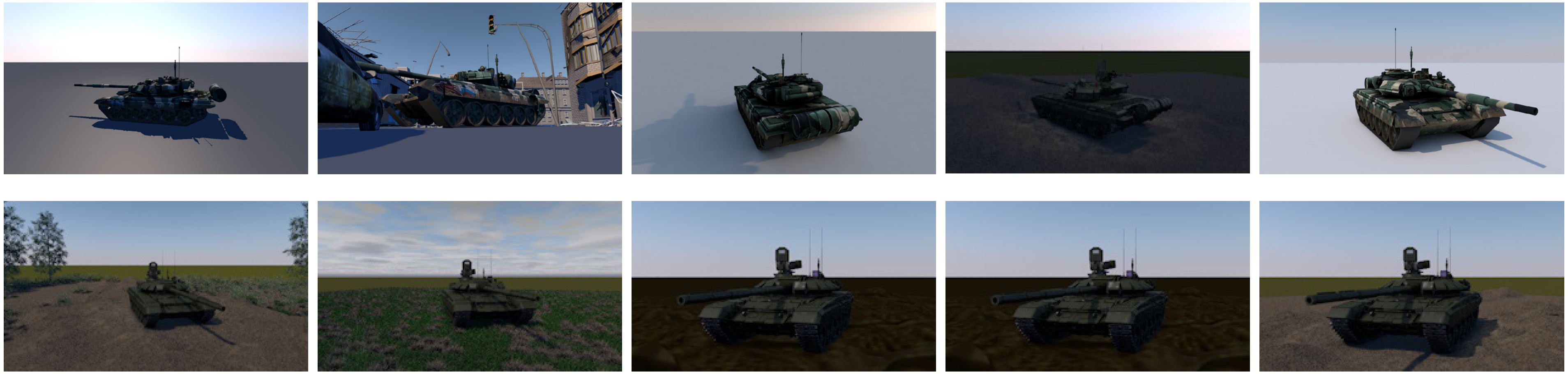



**Battle-Cloner_Controls:** This uses a Mograph Cloner which can clone any object using different arrays. The 'battle' cloner is set to array the tanks linearly whilst using various value generators to control the number of tanks cloned, their rotation, spacing and distribution. This attempts to simulate tanks in battle, more haphazardly placed across the landscape.

**Convoy-Cloner_Controls:** The same cloner here keeps the tanks arrayed in a linear formation, like a convoy moving along a route or in a parade. The visibility of this convoy can be varied to toggle on and off.

# 4.0 RENDER BATCHES

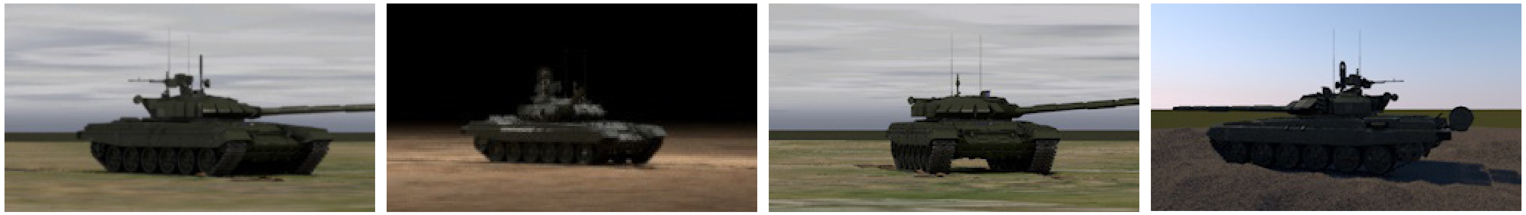


**01. Variable-Control Tests**

1000 images per batch, in which we were testing the variable controls (camera, environment, object) and their optimisation.

**02. Render-Setting Tests**

1 image per render setting in which we tested Cinema 4D (R19)'s 3 native renderers: Standard, Physical and ProRender.



**03. Render-Setting Batch Tests**

Batches of 1000 using best and worst render-settings.

# 5.0 CINEMA 4D RENDER TESTS

| | 163x250_real-image | image-name | classifier-score |
|---|---|---|---|
| **00_Baseline-Image** |  | **00**.tank_133._t-72b3_-_tankbiathlon2013-09.jpg | **98.51%** |

**We focused our following tests on the impact different render settings may have on the quality of the synthetic images, render times per 1 frame, 1000 and 5000 frames, and how render settings affect the score of the classifier.**

For these tests we decide to pick one real-world image of a tank that scores quite well and use it as a bench-mark score for our tests.

Above, the real-world image contains a clearly identifiable tank in a generic landscape. It scores 98.51% by using a pre-trained classifier (MobileNet based on the ImageNet 1000 class label dataset).

We simulate the content of the real-world image and generate single images with optimised render settings for each, running them through the same pre-trained classifier and noting the scores.

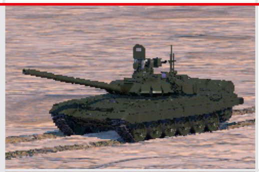
In the pages to follow we conduct comparative tests between images on specific render setting tweaks, such as the image's dpi, Anti-Aliasing filters, toggling Global Illumination and/or Ambient Occlusion on and off etc. All these whilst keeping the rest of the settings at default.

Cinema 4D R19 has 3 native renderers, Standard, Physical and ProRender (GPU renderer). We run our tests through these.

## 5.1 STANDARD RENDERER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **01_DPI_Anti-Aliasing[Best]** |  | **01**.CD4-Standard_AA[Best]_72dpi | 00:00:29 | 8 hrs 3 mins | 1 day 16 hrs 15 mins | **97.39 %** |
| |  | **02**.CD4-Standard_AA[Best]_150dpi | 00:00:27 | 7 hrs 30 mins | 1 day 13 hrs 30 mins | **98.27 %** |
| |  | **03**.CD4-Standard_AA[Best]_250dpi | 00:00:29 | 8 hrs 3 mins | 1 day 16 hrs 15 mins | **98.41 %** |
| |  | **04**.CD4-Standard_AA[Best]_300dpi | 00:00:29 | 8 hrs 3 mins | 1 day 16 hrs 15 mins | **98.91 %** |
| |  | **05**.CD4-Standard_AA[Best]_600dpi | 00:00:27 | 7 hrs 30 mins | 1 day 13 hrs 30 mins | **98.64 %** |
| |  | **06**.CD4-Standard_AA[Best]_1500dpi | 00:00:27 | 7 hrs 30 mins | 1 day 13 hrs 30 mins | **98.04 %** |

## 5.1 STANDARD RENDERER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **02_Anti-Aliasing** |  | **07**.CD4-Standard_AA[None] | 00:00:26 | 7 hrs 13 mins | 1 day 12 hrs 6 mins | **72.43 %** |
| |  | **08**.CD4-Standard_AA[Geometry] | 00:00:27 | 7 hrs 30 mins | 1 day 13 hrs 30 mins | **99.07 %** |
| **03_Ambient-Occlusion** |  | **09**.CD4-Standard_AA[Geometry]+AO | 00:00:26 | 7 hrs 13 mins | 1 day 12 hrs 6 mins | **98.52 %** |
| |  | **10**.CD4-Standard_AA[Best]+AO | 00:00:32 | 8 hrs 53 mins | 1 day 20 hrs 27 mins | **98.97 %** |

# 5.1 STANDARD RENDERER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| 04_Global-Illumination_Anti-Aliasing[Geometry] |  | **11**.CD4-Standard_GI[Default]_AA[Geometry] | 00:00:30 | 8 hrs 20 mins | 1 day 17 hrs 40 mins | **98.52 %** |
| |  | **12**.CD4-Standard_GI[Default]_AA[Geometry]_AO | 00:00:30 | 8 hrs 20 mins | 1 day 17 hrs 40 mins | **98.65 %** |
| |  | **13**.CD4-Standard_GI[Physical-Sky]_AA[Geometry] | 00:00:36 | 10 hrs | 2 days 2 hrs | **97.27 %** |
| |  | **14**.CD4-Standard_GI[Physical-Sky]_AA[Geometry]_AO | 00:00:36 | 10 hrs | 2 days 2 hrs | **98.81 %** |

## 5.1 STANDARD RENDERER

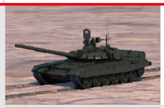
| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| 05_Global-Illumination_Anti-Aliasing[Best] |  | **15**.CD4-Standard_GI[Default]_AA[Best] | 00:00:37 | 10 hrs 17 mins | 2 days 3 hrs 23 mins | **97.90 %** |
| |  | **16**.CD4-Standard_GI[Default]_AA[Best]_AO | 00:00:41 | 11 hrs 23 mins | 2 days 37 hrs 22 mins | **98.44 %** |
| |  | **17**.CD4-Standard_GI[Physical-Sky]_AA[Best] | 00:00:43 | 11 hrs 57 mins | 2 days 11 hrs 43 mins | **98.41 %** |
| |  | **18**.CD4-Standard_GI[Physical-Sky]_AA[Best]_AO | 00:00:48 | 13 hrs 20 mins | 2 days 18 hrs 40 mins | **99.21 %** |

## 5.2 PHYSICAL RENDERER

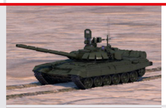
| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **01_DPI_Anti-Aliasing[Best]** |  | **01**.CD4-Physical_AA[Cubic]_72dpi | 00:00:53 | 14 hrs 43 mins | 3 days 1 hrs 36 mins | **98.41 %** |
| |  | **02**.CD4-Physical_AA[Cubic]_150dpi | 00:00:57 | 15 hrs 50 mins | 3 days 7 hrs 10 mins | **97.75 %** |
| |  | **03**.CD4-Physical_AA[Cubic]_250dpi | 00:00:53 | 14 hrs 43 mins | 3 days 1 hrs 36 mins | **98.14 %** |
| |  | **04**.CD4-Physical_AA[Cubic]_300dpi | 00:00:53 | 14 hrs 43 mins | 3 days 1 hrs 36 mins | **98.44 %** |
| |  | **05**.CD4-Physical_AA[Cubic]_600dpi | 00:00:54 | 15 hrs | 3 days 3 hrs | **98.44 %** |
| |  | **06**.CD4-Physical_AA[Cubic]_1500dpi | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **98.69 %** |

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **02_Anti-Aliasing** |  | **04**.CD4-Physical_AA[Cubic] | 00:00:53 | 14 hrs 43 mins | 3 days 1 hrs 36 mins | **98.44 %** |
| **02_Anti-Aliasing** |  | **07**.CD4-Physical_AA[Gauss] | 00:00:56 | 15 hrs 33 mins | 3 days 5 hrs 46 mins | **99.09 %** |
| **02_Anti-Aliasing** |  | **08**.CD4-Physical_AA[Mitchell] | 00:00:53 | 14 hrs 43 mins | 3 days 1 hrs 36 mins | **98.96 %** |
| **02_Anti-Aliasing** |  | **09**.CD4-Physical_AA[Sinc] | 00:00:54 | 15 hrs | 3 days 3 hrs | **98.04 %** |
| **02_Anti-Aliasing** |  | **10**.CD4-Physical_AA[Box] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **99.19 %** |
| **02_Anti-Aliasing** |  | **11**.CD4-Physical_AA[Triangle] | 00:00:54 | 15 hrs | 3 days 3 hrs | **99.06 %** |
| **02_Anti-Aliasing** |  | **12**.CD4-Physical_AA[Catmull] | 00:00:54 | 15 hrs | 3 days 3 hrs | **99.33 %** |
| **02_Anti-Aliasing** |  | **13**.CD4-Physical_AA[PAL] | 00:00:52 | 14 hrs 26 mins | 3 days 13 mins | **99.19 %** |

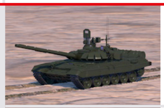
## 5.2 PHYSICAL RENDERER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **03_Ambient-Occlusion** |  | **14**.CD4-Physical_AO[Default]_AA[Cubic] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **98.09 %** |
| |  | **15**.CD4-Physical_AO[Default]_AA[Gauss] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **97.39 %** |
| |  | **16**.CD4-Physical_AO[Default]_AA[Mitchell] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **98.31 %** |
| |  | **17**.CD4-Physical_AO[Default]_AA[Sinc] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **98.51 %** |
| |  | **18**.CD4-Physical_AO[Default]_AA[Box] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **97.69 %** |
| |  | **19**.CD4-Physical_AO[Default]_AA[Triangle] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **97.55 %** |
| |  | **20**.CD4-Physical_AO[Default]_AA[Catmull] | 00:00:56 | 15 hrs 33 mins | 3 days 5 hrs 46 mins | **97.46 %** |
| |  | **21**.CD4-Physical_AO[Default]_AA[PAL] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **97.72 %** |

04_Global-Illumination_Anti-Aliasing_Ambient-Occlusion

| 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|
|  | **22**.CD4-Physical_GI[Default]_AA[Cubic] | 00:00:55 | 15 hrs 16 mins | 3 days 4 hrs 23 mins | **98.01 %** |
|  | **23**.CD4-Physical_GI[Physical-Sky]_AA[Cubic] | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **97.92 %** |
|  | **24**.CD4-Physical_GI[Default]_AA[Cubic]_AO | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **98.97 %** |
|  | **25**.CD4-Physical_GI[Default]_AA[Gauss] _AO | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **98.38 %** |
|  | **26**.CD4-Physical_GI[Default]_AA[Mitchell] _AO | 00:01:01 | 16hrs 57 mins | 3 days 12 hrs 43 mins | **99.18 %** |
|  | **27**.CD4-Physical_GI[Default]_AA[Sinc] _AO | 00:00:56 | 15 hrs 33 mins | 3 days 5 hrs 46 mins | **99.00 %** |
|  | **28**.CD4-Physical_GI[Default]_AA[Box] _AO | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **99.19 %** |
|  | **29**.CD4-Physical_GI[Default]_AA[Triangle] _AO | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **99.19 %** |
|  | **30**.CD4-Physical_GI[Default]_AA[Catmull] _AO | 00:00:59 | 16 hrs 23 mins | 3 days 9 hrs 57 mins | **99.20 %** |
|  | **31**.CD4-Physical_GI[Default]_AA[PAL] _AO | 00:00:58 | 16 hrs 7 mins | 3 days 8 hrs 33 mins | **96.99 %** |
|  | **32**.CD4-Physical_GI[Physical-Sky]_AA[Cubic]_AO | 00:01:02 | 17 hrs 13 mins | 3 days 14 hrs 6 mins | **99.01 %** |

## 5.2 PHYSICAL RENDERER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **05_Physical-Render_Optimised** |  | **33**.CD4-Physical_Optimised | 00:01:10 | 19 hrs 27 mins | 4 days 1 hr 13 mins | **97.18 %** |
| |  | **34**.CD4-Physical_Optimised_AO | 00:01:20 | 22 hrs 13 mins | 4 days 15 hrs 7 mins | **97.13 %** |
| |  | **35**.CD4-Physical_Optimised_GI[Optimised] | 00:03:15 | 2 days 6 hrs 10 mins | 11 days 6 hrs 50 mins | **98.54 %** |
| |  | **36**.CD4-Physical_Optimised_GI[Physical-Sky_Custom] | 00:04:48 | 3 days 8 hrs | 16 days 16 hrs | **97.80 %** |
| |  | **37**.CD4-Physical_Optimised_GI[Optimised]_AO | 00:03:41 | 2 days 33 mins | 12 days 18 hrs 57 mins | **98.75 %** |
| |  | **38**.CD4-Physical_Optimised_GI[Physical-Sky_Custom]_AO | 00:05:02 | 3 days 11 hrs 20 mins | 17 days 8 hrs 40 mins | **98.68 %** |

## 5.2 PRORENDER

| | 163x250_image | image_reference | render_time/ 1f | render_time/ 1000f | render_time/ 5000f | classifier_score |
|---|---|---|---|---|---|---|
| **01_DPI_Anti-Aliasing[Best]** |  | **01**.CD4-Pro-Render_AAS[0-Gauss] | 00:06:04 | 4 days 4 hrs | 20 days 20 hrs | **95.91 %** |
| |  | **02**.CD4-Pro-Render_AAS[1-Gauss] | 00:05:50 | 4 days 1 hr 13 mins | 20 days 6 hrs 6 mins | **99.46 %** |
| |  | **03**.CD4-Pro-Render_AAS[2-Gauss] | 00:09:55 | 6 days 21 hrs 17 mins | 34 days 10 hrs 23 mins | **99.37 %** |
| |  | **04**.CD4-Pro-Render_AAS[3-Gauss] | 00:13:49 | 9 days 14 hrs 17 mins | 47 days 23 hrs 23 mins | **99.20 %** |
| |  | **05**.CD4-Pro-Render_AAS[5-Gauss] | 00:21:48 | 15 days 3 hrs 20 mins | 75 days 16 hrs 40 mins | **99.32 %** |

# 6.0 FUTURE DIRECTIONS

**Optimisation and Further Testing:**

Even in our Cinema 4D pipeline there are a lot of tests to be furthered. For instance:

- The impact of the relationship between focal length and distance from object.
- Render setting tests to be done across multiple images of tanks (with averaging of classification scores to determine lowest variance and highest probability of success)
- Variation of object's texture
- Impact of camera's depth-of-field
- Motion blur and simulations closer to the quality of you-tube video stills. This is due to our interest in analysing images extracted from you-tube videos.
- Test images at higher resolutions, since our tests so far have been on images at 250x144 pixels (since the pre-trained classifier we used only used such lower resolutions).

**Other Software:**

We are yet to test the effects of different software and other renderers on the image output.

**Detection:**

Another area to automate is the labelling of the objects of interest in the synthetic images generated. This can be useful for training classifiers to conduct object-detection rather than just generalised classification.

**Training:**

We are yet to demonstrate the effects of training a classifier on the synthetic data we have generated. Possible training data could include purely synthetic datasets or a mixture of synthetic and real-world. It would be interesting to even test the training of synthetic data in the deeper layers of a neural net whilst training outer layers on real-world data (and vice-versa).

**Post-Processing:**

We are interested in other methods that can alter the final synthetic output further so as to simulate the effects of real-world images better. For instance image-compression, colour aberration etc., which could be possible through the use of Generative Adversarial Networks (GANs).

Set-up in Unity

Set-up in Blender (Using Cycles Renderer)

# 7.0 RESOURCES + READINGS

- **The Driver's Assistant: Utilising Synthetic Data Generation and Deep Learning for Traffic Sign Classification:** https://github.com/alexandrosstergiou/Traffic-Sign-Recognition-basd-on-Synthesised-Training-Data
- **Learning from Simulated and Unsupervised Images through Adversarial Training:** https://arxiv.org/pdf/1612.07828.pdf
- **Synthetic Face Meshes:** https://github.com/PvtKowalski/synthetic-faces
- **Using Synthetic Data for Deep Learning Video Recognition:** https://medium.com/twentybn/using-synthetic-data-for-deep-learning-video-recognition-49be108a9346
- **The Event-Camera Dataset and Simulator: Event-based Data for Pose Estimation, Visual Odometry, and SLAM:**
    - https://github.com/uzh-rpg/rpg_davis_simulator
    - https://arxiv.org/pdf/1610.08336.pdf
    - https://www.youtube.com/watch?v=bVVBTQ7l36I&feature=youtu.be
- **Benefit of Large Field-of-View Cameras for Visual Odometry:**
    - https://github.com/uzh-rpg/rpg_davis_simulator
    - https://arxiv.org/pdf/1610.08336.pdf
- **Synthetic Datasets for training AI (more for extra references):** http://www.immersivelimit.com/blog/synthetic-datasets-for-training-ai
- **Towards End-to-end Text Spotting with Convolutional Recurrent Neural Networks:** https://arxiv.org/abs/1707.03985
- **Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization:** https://arxiv.org/abs/1804.06516
- **Object Detection Using Deep CNNs Trained on Synthetic Images:** https://arxiv.org/pdf/1706.06782.pdf
- **Trained 3D Models for CNN based Object Recognition:** https://www.researchgate.net/publication/315857394_Trained_3D_Models_for_CNN_based_Object_Recognition

Forensic
Architecture