**Cinema "simple" Database Specification v1.0**

**October 10, 2014**

**David Rogers, Jim Ahrens, John Patchett**

**Overview**

A Cinema database is a metadata-enriched set of images used to explore complex data.

This document describes the first release of the Cinema Database, and is intended as instructions to write or read data in a standard Cinema Database format. This initial implementation includes specifics for a file-based implementation, but the schema described is implementation agnostic.

**Use cases**

The Cinema Database supports the following three use cases:

1. Searching/querying of meta-data and images. Samples can be searched purely on metadata, on image content, onposition, on time, or on a combination of all of these.
2. Interactive visualization of sets of images.
3. Playing interactive visualizations, allowing the user on/off control of elements in the visualization.

**Notes on Specification v1.0**

* Cinema Database "simple" version "1.0" supports only images as data samples.
* Labels
  + All labels can contain any Basic Latin Unicode character except for the slash ('/') and null characters ('\0').
  + All database objects must have labels that are unique among instances of that object. In other words, all **collections** have unique labels, etc.

**Database**

A database consists of:

* Required
  + key,value pairs
    - CinemaDB type = "simple"
    - CinemaDB version = "1.0"
    - CinemaDB label = <user defined>
  + One or more **collections**.

**Collection**

A **collection** consists of:

* Required
  + A single **camera**
  + A single set of **timevalues**
  + A set of one or more **samples**

**Camera**

A camera defines a set of sample points. These, in addition to the timevalues, define all possible samples.

* This version of the specification supports only a spherical camera, representing images taken from a point on a sphere directed towards the center of that sphere. The points on the sphere are defined by a regular phi and theta sampling of that sphere.
* Required:
  + A list of values for phi. This is a list of string values having no duplicates. Phi can vary from 0.0 to 180.0, with 0.0 being defined as the 'south pole' and 180.0 as the 'north pole'.
  + A list of values for theta. This is a list of string values having no duplicates. Theta varies from 0.0 to 360.0.
  + Integer values for the camera's [width, height] in pixels.

**Time values**

A set of unique time values.

* Required:
  + A list of values. This is a list of string values having no duplicates. The interval between values need not be the same.

**Sample**

A sample is a set of images matching a specific **camera**.

* Required
  + "label" key, value pair
  + A single image for each (theta, phi, timestep) triplet. The file formate of the image should be a common format such as jpeg, tiff, png, etc.

**JSON schema**

This schema encodes the information for the above specification, and contains the required information needed to create a specific instance of a Cinema "simple" Database.

{

"CinemaDB": {

"label": <string>,

"type" : "simple",

"version" : "1.0",

"collections": [

{

"label" : <string>,

"time": {

"values" : <list of string values>

},

"camera": {

"label": <string>,

"theta": {

"values" : <list of string values>

},

"phi": {

"values" : <list of string values>

},

"image": {

"width" : <integer>,

"height": <integer>

}

},

"samples": [

{

"label": <string>

}

]

}

]

}

}

**File based implimentation**

This is the expected implementation for data transfer. There are required files and directory structure, based on the specification. Required directories and files are noted. The contents of each file is detailed below.

required, user-specified <CinemaDB\_label>/

required directory collections/

required file meta.json

required, user-specified <collection\_label>/

required file camera.json

required file time.json

required directory samples/

required, user-specified label <sample\_name>/

required directory image/

required, user-specified label <timestep\_value>/

required, user-specified name <theta\_phi>.<extension>

...

.../

<timestep\_value>/

**File: meta.json**

The contents of this file are as define below for this type and version of the Cinema Database.

{

"type" : "simple",

"version" : "1.0"

}

**File: camera.json**

Contains the values for the camera as noted below.

{

"label" : <string>,

"theta": {

"values" : <list of string values>

},

"phi": {

"values" : <list of string values>

},

"image": {

"width" : <integer>,

"height": <integer>

}

}

**File: time.json**

Contains values for time, as noted below.

{

"values" : <list of string values>

}

**The image/ subdirectory**

The image/ subdirectory contains all images for all (theta, phi, time) triplets. In the subdirectory, there is a directory for each time value, using each value in the list of time values. Within each of these directories, there are (number of theta values)\*(number of phi values) images of some common image format. Note that no ordering is enforced, so the timestep\_value/ directories will be ordered by the filesystem.

image/

<timestep\_value>/

<theta\_phi>.<extension>

...

**Simple Example**

This example is based on above a JSON schema, and shows the specific contents of the required files, and the names that result from the time, theta and phi values.

{

"CinemaDB": {

"label": "halo\_sim",

"type" : "simple",

"version" : "1.0",

"collections": [

{

"label" : "earth",

"time": {

"values" : ["0.0", "0.1", "0.2"]

},

"camera": {

"label": "global\_camera",

"theta": {

"values" : ["0.0", "90.0", "180.0", "270.0"]

},

"phi": {

"values" : ["0.0", "45.0", "90.0", "135.0"]

},

"image": {

"width" : 1920,

"height": 1080

}

},

"samples": [

{

"label": "salinity"

}

]

}

]

}

}

**Directories and files written to disk**

halo\_sim/

collections/

earth/

camera.json

samples/

salinity/

image/

0.0/

0.0\_0.0.png

0.0\_45.0.png

0.0\_90.0.png

0.0\_135.0.png

90.0\_0.0.png

90.0\_45.0.png

90.0\_90.0.png

90.0\_135.0.png

180.0\_0.0.png

180.0\_45.0.png

180.0\_90.0.png

180.0\_135.0.png

270.0\_0.0.png

270.0\_45.0.png

270.0\_90.0.png

270.0\_135.0.png

0.1/

(identical to contents of 0.0/)

0.2/

(identical to contents of 0.0/)

time.json

meta.json

**File: meta.json**

{

"type" : "simple",

"version" : "1.0"

}

**File: camera.json**

{

"label" : "global\_camera"

"theta": {

"values" : ["0.0", "90.0", "180.0", "270.0"]

},

"phi": {

"values" : ["0.0", "45.0", "90.0", "135.0"]

},

"image": {

"width" : 1920,

"height": 1080

}

}

**File: time.json**

{

"values" : ["0.0", "0.1", "0.2"]

}