C-SPACE
An affordable tool to turn the space surrounding us into a creative experience

Deliverable D5.3.1

Assessment report and recommendation for further developments 1

WP 5 – Trials and Validation of demonstrator

T.5.4 - Recommendations

Revision: [FINAL]

Authors:
Oliver Posniak (Fraunhofer)
Martin Ritz (Fraunhofer IGD)

Dissemination level	PU (public)
Contributor(s)	Andrés Navarro (Vicomtech) Anestis Trypitsidis (EPSILON) Bruno Simões (Graphitech) Martin Ritz (Fraunhofer IGD) Petr Aksenov (TU/e)
Editor(s)	Oliver Posniak (Fraunhofer) Bruno Simões (Graphitech) Roderic Molina (GISIG)
Partner in charge(s)	GISIG, Fraunhofer
Due date	31-Dec-14
Submission Date	28-Jan-15

Document History

1. Revision History

Revision Number	Revision Date	Summary of Changes	Author
0	09/12/14	First table of content	Roderic Molina (GISIG)
1	17/12/14	First draft	Roderic Molina (GISIG)
2	26/01/15	First combined draft	Martin Ritz (Fraunhofer)
3	28/01/15	Combined draft	Oliver Posniak (Fraunhofer)

2. Reference Documents

Please see the following documents for more information:

Document Name	Version	Author
D2.1 - First version of the 4D reconstruction pipeline	NA	c-Space
D3.1 - EGNOS SiSNet Service Connectivity	NA	c-Space
D3.4 Geotagging Software Module	NA	c-Space
D3.6 Initial version of the Affective module	NA	c-Space
D4.2 Content Access Infrastructure documentation	NA	c-Space
D5.2.1 Deployment Plan 1	NA	c-Space
D5.1 Evaluation Methodology	NA	c-Space

Table of content

Acronyms

CA	Consortium Agreement
QRM	Quality and Risk Manager
GA	Grant Agreement
PC	Project Coordinator
TB	Technical Board
TL	Task Leader
WPL	Work Package Leader
GRAPHITECH	Fondazione Graphitech
VICOMTECH	Fundacion Centro de Tecnologias de Interaccion Visual y Comunicaciones Vicomtech
GISIG	GISIG - Geographical Information Systems International Group Associazione
Amitié	AMITIE SRL
FRAUNHOFER	Fraunhofer-Gesellschaft Zur Foerderung Der Angewandten Forschung E.V
TU/e	Technische Universiteit Eindhoven
EPSILON INTERNATIONAL	Epsilon Internasional Anonymi Etaireia Meleton Kai Symvoulon (Epsilon International Sa)
TECHNOPORT	TECHNOPORT SA

1 Introduction

A consolidated system is only as good as its individual components. This is also true in the case of c-Space. During the first year a number of developments were launched to start creating and testing individual components which would have to work together in a future c-Space platform. Among these were the server side services such as the recommender service, the 3D/4D reconstruction service to name a few. On the client side were modules for geotagging, augmented location, augmented reality, affective computing, routing and social mechanisms.

This document describes the current development status of the initial most important components being developed in c-Space by assessing open and solved issues and their current performance.

1.1 Objectives

The main objectives of the first assessment round are to identify:

- The current development status of the individual components
- Solved and open issues by testing components against real scenarios
- Giving recommendations for further development

2 Data assessment from Pilots

There are three pilots in c-Space and to optimize and parallelize development of crucial components, we decided to associate each of them to a pilot and a corresponding team of partners, sometimes taking into account the partner's geographical proximity as an advantage for collaborative work.

Therefore we developed and assessed the recommender system with the cultural heritage pilot and in particular the city of Bologna. The basic 3D reconstruction capabilities we needed as a foundation for the subsequent 4D reconstruction, we tested with the architecture and urban planning scenario, in particular the Begato fort in Genova. Finally, the first development steps towards a 4D reconstruction were tested with a subset of feasible products and concepts in the advertising scenario in Luxembourg.

2.1 Cultural Heritage and Personalized Tourism

This section discusses the data collections that have been created, processed, analyzed and fine-tuned with respect to the Bologna pilot on cultural heritage and personalized tourism. The requirements on and the planned usage intentions of the data have been taken into account for this purpose.

Several data collections have been produced in accordance with the description and the details of running the pilot:

- **Data on points of interest (POIs)** in Bologna. The generated categories are the following:

- *A custom, place-specific vocabulary* used to categorizing a POI, which is available for including in a tour, into one or more groups. The content of this vocabulary has been the result of the analysis of the available POIs in the city of Bologna, and the development of the vocabulary has undergone a number of iterations from the point of view of the envisioned recommendation process. The categories produced this way become the base for the interests and preferences for categories specified by the users at the tour planning and creation times, which assures the alignment between the two parts (vocabulary and users' preferences) that will be used during the recommendation process.
- *A number of collections of POIs*. At the moment, the following categories have been produced: *"Museums, Libraries, Archives", "Heritage", "Entertainment",* and *"Sport"*. Each collection's schema consists of a corresponding set of the details about a POI. Several iterations of data completion and re-organization have taken place, in accordance with the posed requirements and availability of information about the POIs. Where necessary, additional input not envisioned initially, was provided. The details include two groups of data:
  - *POI-oriented details*, such as its name, location on the map, categorization (as per the established vocabulary), importance among all the POIs at that destination, etc.;
  - More specific *user-oriented details*, such as matching to the needs satisfaction, opening hours, entrance fees, etc.

- *Details of the local Transportation system*, including public transport, information on parking, and availability and use of digital maps.

- **Data to describe the users** of the cultural tourism component. This comes in four main categories:
  - *General data about the users*, such as their demographics and background.
  - *Information related to the trip*, such as time scope, main preferences, general state of needs and interests, as well as the main intentions, such as POI information.
  - *Details related to a particular segment of the trip*, such as the start and end time of this segment, the allowances for this segment, as well as the history of and preferences for visitation for this segment.
  - *Details describing the actual state of the user*, including the state of needs to this very moment, the actual affective state, as well as the location and the position within the tour plan.

All aforementioned categories of data have been analyzed and processed in several iterations each. Where necessary, they have been considered in combination so as to allow the alignment between independent sets, and have further been framed according to the needs of the Bologna pilot altogether. Further corrections or amendments to the structure of the above data categories may be possible, as will be appropriate and in accordance with the pilot's contingent details. Such amendments will be, in general, considered as extensions to the existing collections.

2.2 Architecture and Urban Planning

A massive amount of more than 10GB of footage was generated to support evaluation of tests with the prototype in the context of the pilot 'Architecture and Urban Planning'. This data was and will further be used to assess the prototype of the 3D Reconstruction pipeline regarding quality of results, robustness and speed.

On August 7 and 8 in 2014, a drone operation company carried out flights for aerial acquisition of videos and still images covering the premises and perimeter of the Begato fort at Genova. The fort was built in 1823 on a hill at 475 metres altitude North of Genova (see Figure 1). Use of the premises was granted under cooperation agreement between the city of Genoa and GISIG.

Figure 1: Begato Fort, ground plan (left) and aerial view (right).

The fort consists of wide parts of flat land, surrounded by walls of up to 4m height, with some small separate buildings scattered over the area, as well as a central square structure that is mostly intact due to reconstruction, with one of the four corner towers left partly decayed. The edifice features an inner yard of about 20 by 20 meters, the overall building side length (without towers) measures about 45 meters.

As carrier for the photo camera, a custom made quad copter drone was used, equipped with a Panasonic Gm1 camera with 12:32mm optics (see Figure 2).

Figure 2: Drone used (left) and camera setup (right).

Planning the acquisition campaign is a crucial step which is highly dependent on the camera and lens used as well as the height of the respective object to be captured. For successful photogrammetric reconstruction using the method chosen for c-Space, an overlap of at least 80% between neighbouring camera perspectives is to be achieved, which means that 80% of the parts of the scene visible in one shot need to be visible in the neighbouring perspectives as well, while 20% of new content can come in. Thus, a trade-off has to be met between too little overlap on one end, and as a consequence potentially insufficient common feature points, leading to failure in 3D reconstruction, and too high overlap on the other end, which reduces efficiency and results in a number of mostly redundant images potentially too high for reasonable reconstruction. The height of the object, in this case the outer wall and buildings, combined with the vertical field of view defined by the camera and lens used, determine the number of images that need to be captured as a sequence of neighbouring images in vertical direction, in that the step size of vertical shooting positions relative to each other is computed to meet the overlap constraint.

An analogous calculation taking horizontal field of view into account leads to the step size that needs to be applied for subsequent horizontal capturing positions. Also, the distance between camera and object needs to be defined, again being a trade-off between efficiency and accuracy, since the resolution of photogrammetric 3D reconstructions is directly dependent on the distance of the sensor from the object, taking into account field of view of camera and lens combined. Camera and lens are mentioned as a union here, since the field of view of the overall system depends on the combination of lens and camera sensor.

To help GSIC with trajectory planning, IGD provided some calculations of the theoretic approach to optimal acquisition of the footage around the fort perimeter as well as around the central building and within its inner yard. Figure 3 highlights the respective trajectories (yellow). These are a coarse guide for the drone flight route, and it is apparent that as above trade-off, the distance was set to about 15m between camera (=drone) and object, with the distance to be maintained as constant as possible throughout the trajectory.

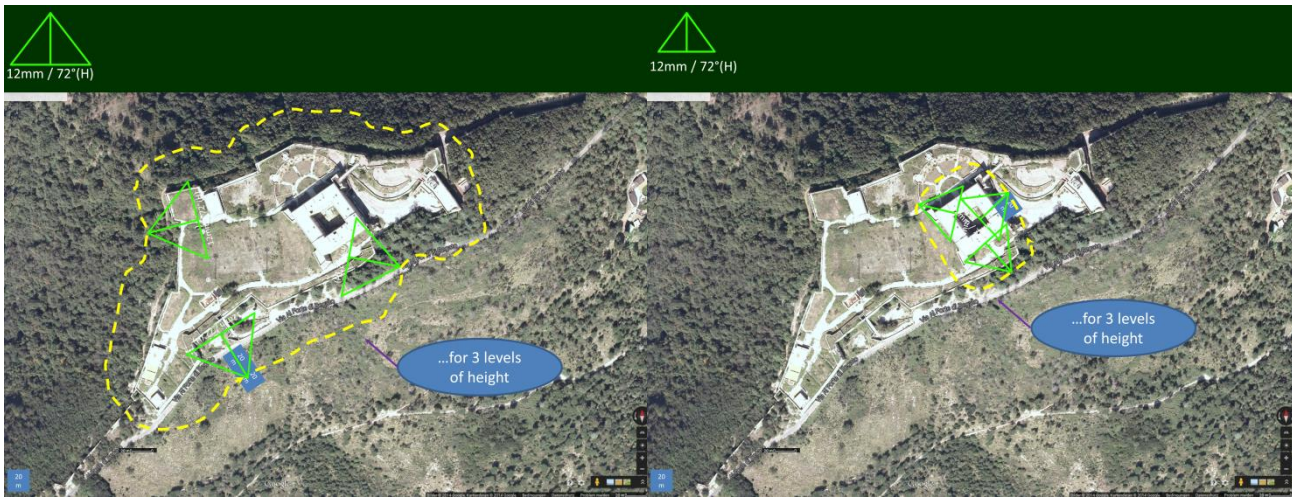

Figure 3: Capturing trajectory for perimeter (left) and around center building (right).

Based on the approximate height of the perimeter walls, repeating all trajectories for three different levels of height was suggested, as indicated in Figure 4 with numbers one through three. The third level was supposed to capture parts of the walls from above as well.

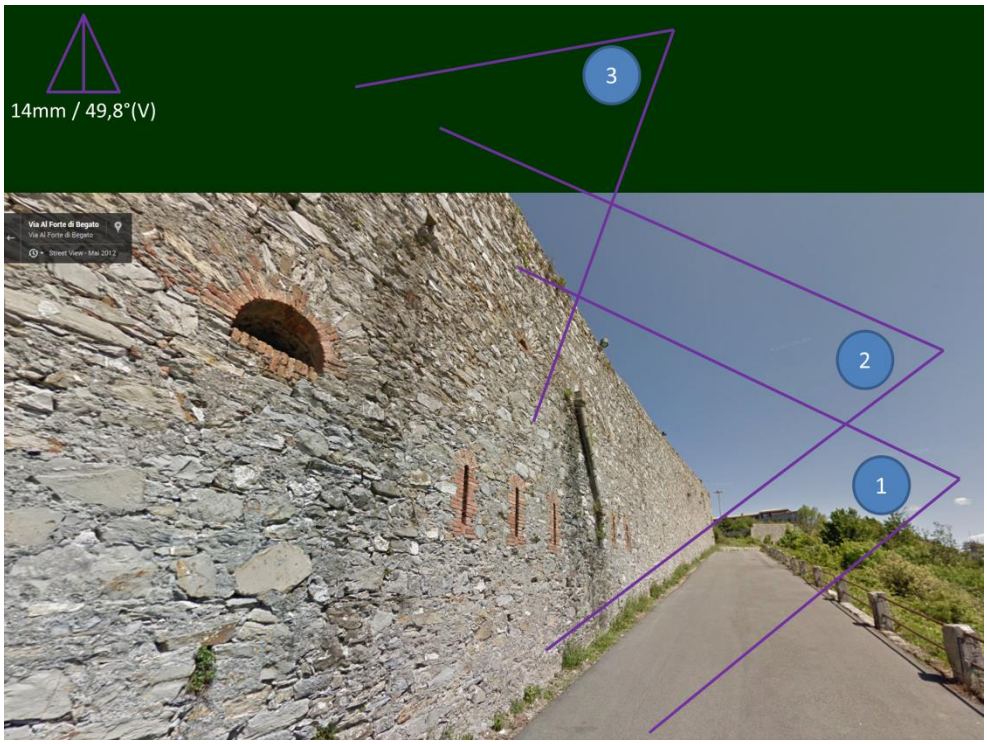

Figure 4: Three different levels of height per trajectory to account for sufficient overlap.

Some example images and videos captured are shown in Figure 5. The footage shown was captured at higher distances, i.e. out of the suggested trajectories, and are presented here as an overview of different parts of the fort. Clearly, the image material, both from videos and still images, is suitable for 3D

reconstruction of a static scene, since the viewpoint varies from image to image, while for 4D reconstruction, several different viewpoints would be needed, capturing the evolution of a scene over time from overlapping perspectives on the scene. Thus, the acquisition campaign was intended for assessment of the first prototype of the 4D model reconstruction pipeline, targeting the first two testing stages as described in 'D5.1: Evaluation Methodology': first, 3D reconstruction of a static scene from still images, and second, from videos that are split into single frames by the module as basis for reconstruction.

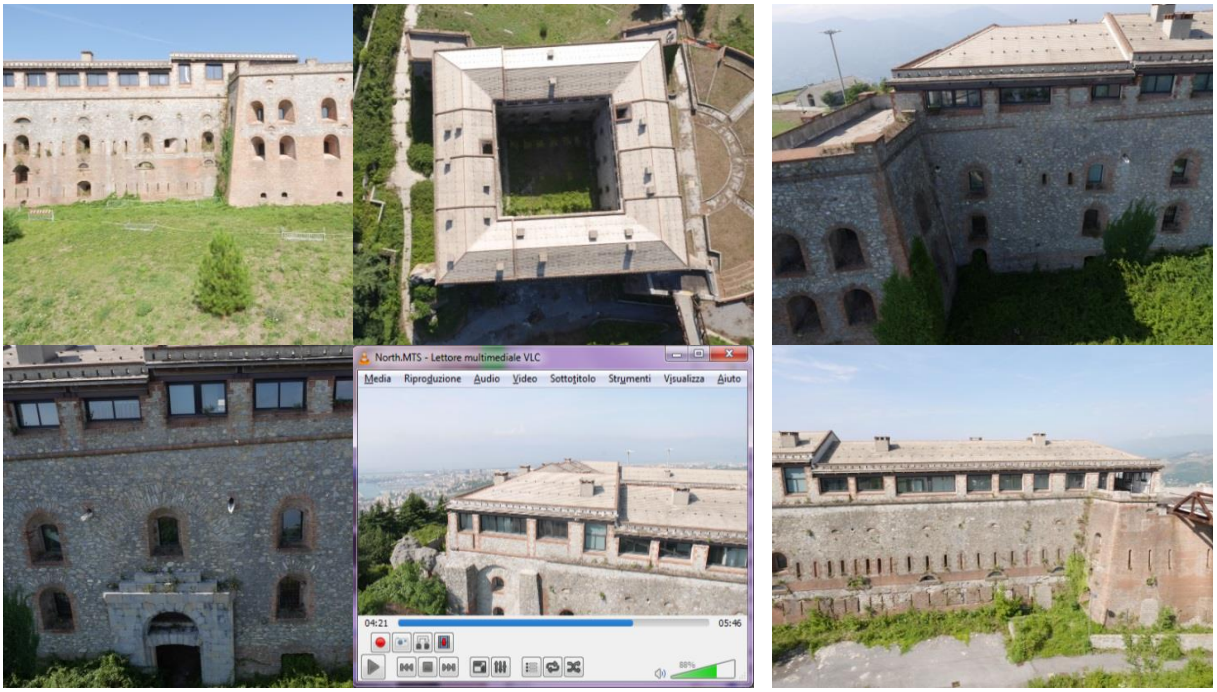

Figure 5: Example images and video captured by camera on drone.

One of the biggest challenges of the project is a 4D reconstruction from videos that are recorded with different video capturing devices. The inputs to the 4D reconstruction module are videos with various resolutions, aspect ratios, file formats and frame rates.

First dynamic test scenario took place at the fortress Begato, where a moving car was recorded with nine various video capturing devices: Android smart phones, iPhones and iPads. Videos were recorded in MPEG-4 (.mp4), Quicktime (.mov) or AVI (.avi) formats, at resolutions of 1920x1080, 1280 x 720, 640 x 480 or 568 x 320 pixels and a frame rate of 30, 29 or 27 fps. Table 1 shows properties of nine videos.

	File format	Resolution [pixels]	Frame rate [fps]	Video length [sec]
Video 1	MPEG-4	1920x1080	29	20
Video 2	Quicktime	1920x1080	29	19
Video 3	MPEG-4	1280x720	30	20
Video 4	MPEG-4	640x480	27	19
Video 5	Quicktime	1920x1080	29	19
Video 6	MPEG-4	1280x720	29	17
Video 7	Quicktime	1280x720	29	19
Video 8	MPEG-4	1920x1080	29	21
Video 9	Quicktime	5658x320	29	19

Table 1: video footage

2.3 Augmented Mobile Advertisement

4D reconstruction and 4D geometry streaming could bring a new facet to advertising. Suppose a new car model is coming out and one could 4D scan the sole concept car or prototype in existence at the factory and then showcase a live test-run through augmented reality replay 1:1 in each of the manufacturer's showrooms across the country to interested potential customers. It is one of many more examples where a dynamically moving object or any dynamic action performed on a product could be digitized in 4D and then visualized perspective-independent from any angle. Because of its immediate market potential the augmented mobile advertisement scenario is one of the first, where our initial 4D digitization approach has been tested on, leading to some very interesting findings as well to improve in our further development of appropriate algorithm and in particular feature point detectors for photogrammetry.

For the mobile advertisement scenario Technoport and Fraunhofer IGD chose potentially suitable products, prototype objects or concepts from a list of companies and contacts:

- **Foobot:** An indoor air monitoring device, helping to set the right temperature and humidity and looking out for toxic gases. (www.foobot.io).
- **AllSquareGolf:** Social network for golfers – 4D Scan of golfer's movements. (www.allsquaregolf.com).
- **Artica:** Robotics workshops for adults and kids during which small robots are created (www.artica.cc).
- **City Mov:** Electric Car sharing (www.citymov.lu).
- **Jamendo:** Music promotion and distribution platform (www.jamendo.com).
- **LuxScan:** Industrial scanners for process automation in the lumber industry.
- **NeoTechpro:** Quadcopter manufacturer
- **OAT:** Testing online tools (http://www.taotesting.com/).
- **Sportunity:** supporting underpriviledged kids in sports (www.sportunity.org).
- **Vibrationmaster:** Vibration testing equipment (www.vibrationmaster.com).

Given the range of different "products" we opted for a first test-run with City Mov, an electric car sharing company in Esch-sur-Alzette, Luxembourg. The electric vehicle is based on a Renault ZOE and initially we

would record it at the company's premises offering a background with many features, ideal for photogrammetry. It would allow us to set up our 12 video cameras in a circle to drive the car around or in a pocket geometry to drive into. However, on last minute notice, the location had to be changed to the upper most park deck of a newly built car park due to the visit of high ranking Luxemburgish government officials.

We chose the car scenario to be the first for the relative simplicity of the object to be scanned in 4D, a rather simple geometry, without too many protrusions and some texture. However, with the change of location and deteriorating weather conditions, we would have to reconstruct a white car on a grey parking deck on a grey raining day, which could have an impact in the quality of the reconstruction and the number of feature correspondences detected between the camera perspectives.

To capture the dynamic scene twelve Qumox cameras on tripods were used (see Figure 6). Positions of cameras were carefully selected in order to ensure sufficient overlap between neighboring cameras. Cameras were positioned in a circle facing the same spot on the park deck, where a dynamic scene was performed.

Figure 6: Esch-sur-Alzette City Mov capture setup preparation and circle with 12 cameras.

In a second attempt we set up a pocket of cameras. All videos were captured in full HD resolution (1920 x 1080 pixels) at 30 frames per second. Video recording length was between 55 and 180 seconds. However, for the 4D reconstruction five seconds of a dynamic scene were selected. The video configuration is shown in the table below (see Table 2):

	File format	Resolution [pixels]	Frame rate [fps]	Video length [sec]
Videos (12x)	Quicktime	1920x1080	30	5

Table 2: Video configuration

Subsequently, on a second occasion, we scanned two additional products, namely a robotic arm (see Figure 7) and small kit-robots (see Figure 8) to be assembled during robotic workshops of Artica.

Figure 7: Esch-sur-Alzette Robotic Arm capture setup

To make sure we would have enough features to find correspondences for in videos from different perspectives, we added markers to the plane on which the robotic arm would stand and to the robot itself. Its recorded task would then be to lift an obstacle and lower it at a different location.

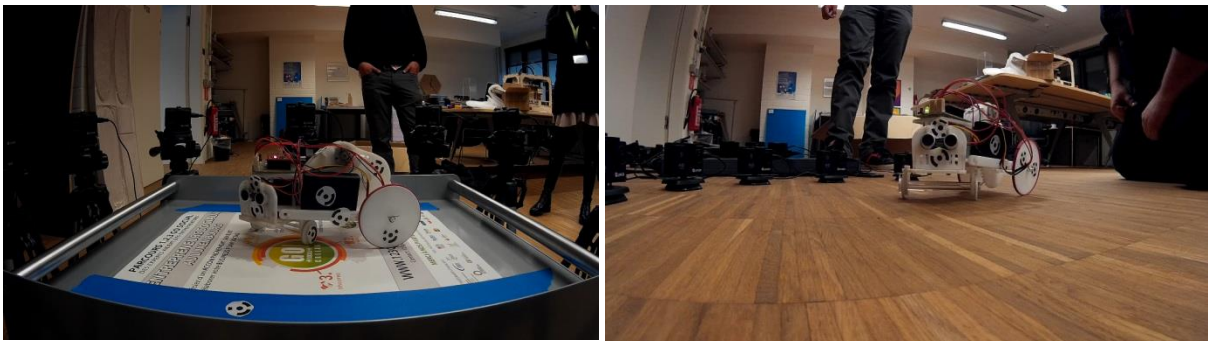

Figure 8: Esch-sur-Alzette Kit-Robot capture setups

We proceeded in analogy with the small, autonomous kit-robots that would back up from an obstacle and change their course using ultrasound sensors for that purpose. Two setups were made. In the first the small robot would move on a cupboard within constraints which would be the side walls/bars. In the second the kit-robot would be placed on the floor as well as the cameras surrounding him (see Figure 8).

3 Assessment & Evaluation Results

In the previous chapter we focused on data acquisition and transformation methods tested with each pilot scenario. In the following sections, we present the assessment results split up by components and sub modules.

The chapter begins with the assessment of the core module. The scene reconstruction pipeline is presented and the results are summarized. This includes 3D as well as 4D reconstruction tests to assess the robustness and quality of the implemented Multi-View Stereo approach. Additionally the testing tools used in this assessment are presented. Finally recommendations for the further development are given based on the results.

To conclude, we provide an evaluation of the mobile application both as a whole and by module. It includes a benchmark of its access to CAI, an assessment of the c-Space geotagging module and an early evaluation of the geometry streaming and visualization framework. We decided to exclude the evaluation of the EGNOS SiSNet Service from this deliverable as results are already documented in the "D.3.1 - EGNOS SiSNet Service Connectivity". Finally recommendations for the further development are also provided.

3.1 4D Pipeline

3.1.1 Overview

The current state of the 4D Pipeline uses images and videos to reconstruct scenes in 3D as well as 4D (e.g. 3D over time). In the first testing cycle several testing scenarios were conducted and used for benchmarking and quality checks of the underlying algorithms. The 4D Pipeline was tested independently from other c-Space modules under controlled conditions by internal developers and employees. The inputs used for testing were selected to be as close as possible to the actual data that will be provided when the c-Space system goes live.

The conducted 3D tests resulted in 3D Models of promising quality. The algorithms used turned out to be very robust in terms of changing conditions and varying overlap between neighboring input images. As mentioned earlier, sufficient image overlap is a significant precondition for successful 3D reconstruction.

While evaluating the 4D Tests, some difficulties were found in terms of robust feature detection and matching. These difficulties will be explained in detail later.

3.1.2 Testing Tools

For testing the performance of the 3D/4D reconstruction pipeline performance measurements are directly implemented in the source code of the pipeline. For visualizing and evaluating the aligned camera positions AgiSoft[1] is used.

The resulting 3D models for both forms of output (3D and 4D models) are displayed with Meshlab[2]. With this software it is possible to display large 3D models with millions of faces.

3.1.3 Test scenarios

To test the 3D / 4D reconstruction pipeline two different groups of tests were conducted with multiple tests in each group. The first group was evaluating the results for 3D reconstruction from images and videos. The second test group focused on the generation of 4D Models (3D models over time) using images and videos from multiple sources and different camera positions to calculate one 3D model for each point in time.

[1] http://www.agisoft.com/
[2] http://meshlab.sourceforge.net/

3.1.3.1 3D Reconstruction

3.1.3.1.1 Framesplitting of videos

To reconstruct 3D-Models from video streams, the video sources have to be split into single frames at first. For this purpose a FrameSplitter was implemented and tested with various video formats.

Tested and supported video formats are:

- mov
- mpg
- avi
- mp4
- mts

The application is able to split every video into the finest possible granularity, i.e., one discrete time step. For example, a 10 second video with 30 frames per second is split into 300 single frames.

3.1.3.1.2 Begato Fort – Images

For the first test of the 3D / 4D reconstruction pipeline still images were taken to evaluate the capability of reconstructing static scenes from still images. For this test the images captured during the drone flights in the course of the test involving the Architecture and Urban Planning pilot were taken into account.

All images captured in the independent drone flights (see Figure 9) were used as input for one single reconstruction of the fort.

Figure 9: Drone aerial photos

Table 3 shows the actual input data for the 3D reconstruction from each drone flight.

	Resolution [pixels]	Number of images
North side	4592*3448	90
East side	4592*3448	172
South side	4592*3448	147
Zenit	4592*3448	65
Interior	4592*3448	64
All	4592*3448	538

Table 3: Input data for 3D reconstruction

Figure 10 shows the first step of a 3D/4D reconstruction process - alignment of multiple images. First, features are detected on all of the images and then matched among image pairs. The goal of the camera alignment process is to reconstruct camera parameters (position, orientation, focal length and radial distortion).

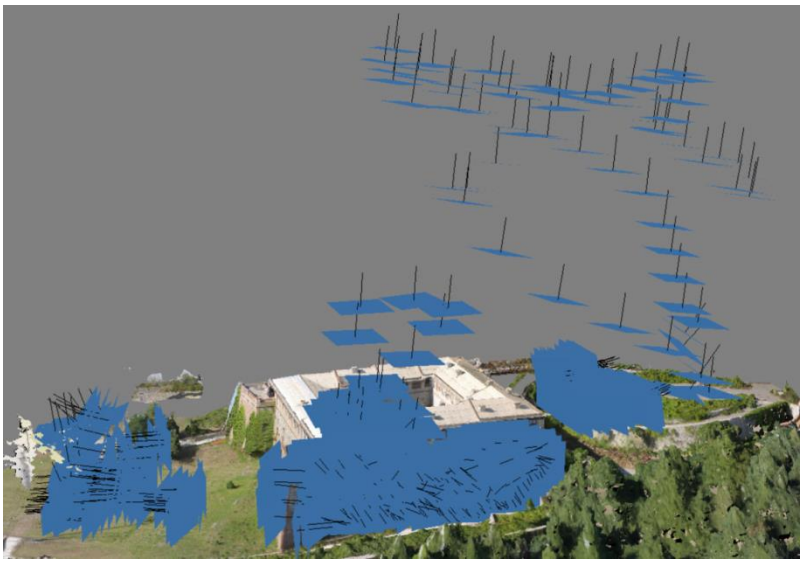

Figure 10: Alignment of All Frames

Based on camera parameters, 3D positions of points are obtained in a Multi-View Stereo algorithm. Resulting dense point cloud is an input for the mesh extraction. Surface is generated and in a last step – texture mapping – a 3D model gets color information.

Figure 11: Resulting 3D model of the Fort

3.1.3.1.3 Begato Fort – Videos

In this test scenario the videos of 6 drone flights were split into single frame sets. The 3D model of each flight was calculated independently to reduce calculation time. First of all the videos were split into frames, and only frames showing the fort included into the input image set for reconstruction, as well as images from the beginning of the drone flight. The stream of frames along the time axis was subsampled in that only every 10th frame of the video streams was picked, resulting in a larger baseline between the single consecutive perspectives, resulting in a bigger triangulation angle (see Figure 12), and leading to a more robust reconstruction.

Figure 12: top: Each frame; bottom: Each 10th image in video

The following Table 4 shows the actual input data for the 3D reconstruction from each video.

	Video		Images		
	Video length [min:sec]	Frames per second	Resolution [pixels]	Number of images	Number of subsampled images
North side	5:46	30	1920*1080	8665	376
East side	6:10	30	1920*1080	9265	259
South side	6:36	30	1920*1080	9901	468
West side	7:33	30	1920*1080	11341	559
Interior	8:46	30	1920*1080	13165	681

Table 4: Input data for 3D reconstruction

The following Figures (see Figure 13 and Figure 14) show results of the 3D reconstruction with detected camera positions and the resulting 3D mesh with mapped texture. In both test scenarios all cameras are aligned properly with a high accuracy of the estimated positions.

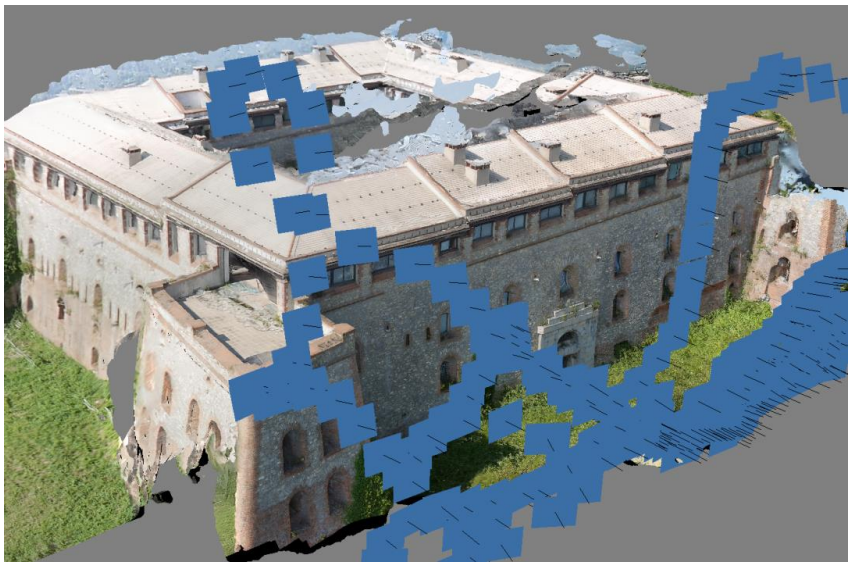

Figure 13: Alignment of East side of the fort

On figures, extrinsic camera parameters (position and orientation) are represented by a black vector line and a blue rectangle at its end[3]. From the visualization of camera positions a drone trajectory can be assumed. Drone trajectories for Begato test scenarios were not consistent. Thus the reconstruction process was laborious. Despite deviations of the drone flight and a huge number of input images the 3D reconstruction pipeline was able to reconstruct the 3D model of a test scenario.

Figure 14: Alignment of interior of the fort

[3] http://uas.trimble.com/sites/default/files/downloads/gatewing_photoscanworkflow090.pdf

More details of the test results regarding the number of aligned images and complexity of the 3D model are provided in sections 3.1.4 and 3.1.5.

3.1.3.2 4D Reconstruction

3.1.3.2.1 Video Synchronization

If multiple users are capturing the same scene, this does not necessarily lead to the videos beginning at the same time. In this case they have to be synchronized. Only with synchronized videos a 3D reconstruction is feasible. At this stage of the development the c-Space application does not have access to a common synchronized time for every user capturing videos. This will be included at a later stage of the development. For this reason the frame synchronization of the 4D Pipeline will be tested later when available. The frames of the videos were split manually to obtain synchronized frame sets.

3.1.3.2.2 Genova – Moving vehicle

First of all, all videos were manually split into single frames (see chapter 3.1.3.1.1). The next challenging task was a time synchronization of frames from various videos. At this stage of the development frame synchronization is not implemented yet, because common time is not available within the c-Space application. Thus, an attempt to synchronize frames manually was performed. At this test scenario there was no significant visual feature in a video on which a specific time point could be defined. The only possibility would be to define a point, when a wheel contacts a specific soil structure on the ground (marked red in Figure 15).

Figure 15: Defining a specific time point in a video

The problem occurred when we tried to define this contact in all views. From some camera positions this point is not visible. Thus, manual synchronization of frames was not possible and the 4D reconstruction was not performed. A 4D reconstruction of this test scenario might be possible as soon as a common time for media is available within c-Space application.

3.1.3.2.3 Luxemburg – Moving vehicle

The second test scenario for 4D reconstruction was a moving car, recorded in Luxembourg. To capture a dynamic scene twelve Qumox cameras on tripods were used. Positions of cameras were carefully selected in order to ensure sufficient overlap between neighboring cameras. All videos were captured in full HD resolution (1920 x 1080 px) at 30 frames per second.

Videos were manually split into single frames and each tenth frame of each video was selected in order to speed up the reconstruction process and to get significant change between sequential 3D reconstructions over time. For easier and more precise manual synchronization of frames from various cameras, a hand-clap in the middle of a scene was performed at the beginning of the videos (see Figure 16).

Figure 16: Frames were manually synchronized based on visual feature

The first step of the 4D reconstruction is alignment of cameras. Eleven out of twelve cameras were aligned correctly (see Figure 17). The alignment was not perfect, because only few features were matched in the middle of a scene. Although a good synchronization of frames was performed and homogenous videos were used as an input for 4D reconstruction, algorithm for matching feature points detected insufficient number of common feature points. The reason for this is that the floor, which covers the largest part of the videos, is reflective and thus the inhomogeneous pattern on the floor is captured from different camera locations.

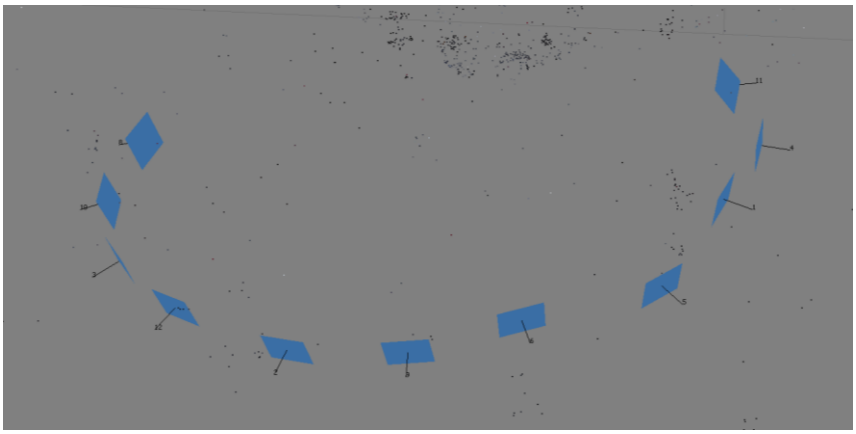

Figure 17: Alignment of cameras

Figure 18 indicates matched points (blue dots) acquired from feature matching algorithm between two neighboring cameras. The footage shows that feature points are matched only in the background.

Figure 18: Matching feature points

Consequently, very sparse dense cloud was reconstructed in a Multi View Stereo algorithm and this is why also the final output of the 4D reconstruction pipeline is not very convincing (Figure 19). In Figure 19 three different timeframes of the 4D reconstruction are shown.

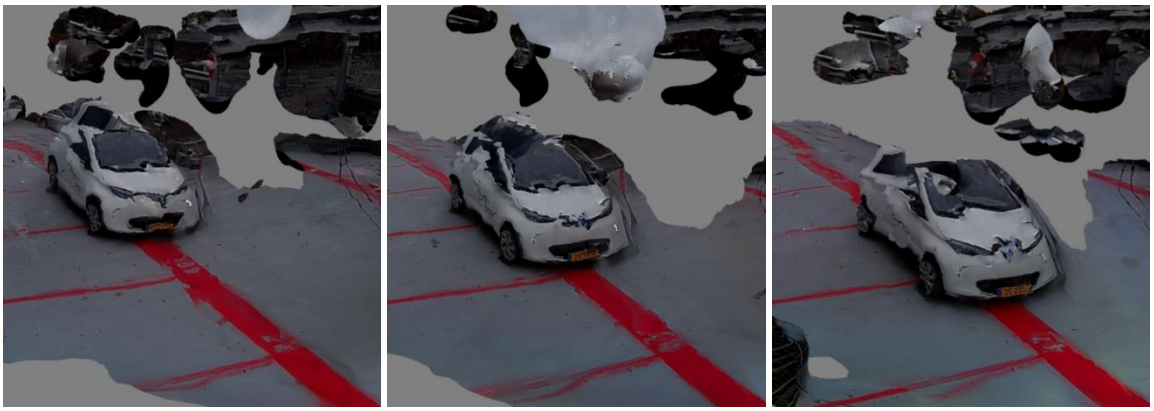

Figure 19: Resulting 4D Reconstruction

3.1.3.2.4 Lab – Christmas moose

Another test scenario for 4D reconstruction was carried out in a controlled environment in a laboratory at Fraunhofer IGD. Twelve Qumox cameras were carefully positioned in two rows in order to provide sufficient overlap and good amount of parallax for a photogrammetric triangulation. Objects in a scene were non-reflective and they had many feature points. Background was stable and lighting conditions were good. Figure 20 shows camera positions of this test scenario.

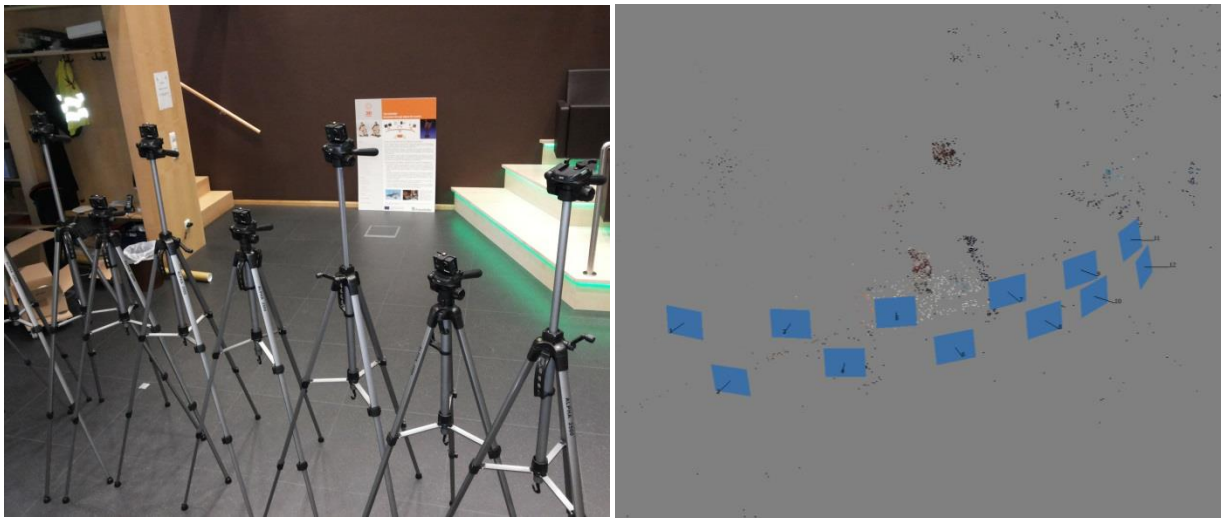

Figure 20: Cameras setup (left) and camera alignment (right)

At the beginning of the scene, a hand clap was performed in order to easily synchronize frames from various videos.

The final result of a 4D reconstruction in a controlled environment (see Figure 21 or https://www.youtube.com/watch?v=IRIaLNdYIJ8) indicates that a good 4D reconstruction is eventually possible. The focus of further development should be on more robust feature matching algorithms that will enable sufficient number of matched feature points even in an unstable environment.

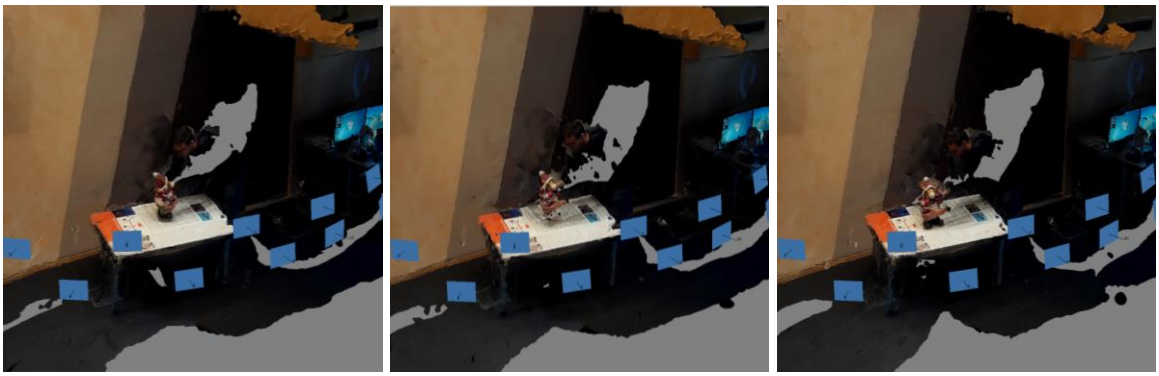

Figure 21: 4D reconstruction in a controlled environment

3.1.3.2.5 Luxemburg – mobile Advertisement's

Another two test scenarios were performed in Luxemburg, at Technoport. A movement of a robotic arm, which picks up an object, and a movement of a small kit-robot were recorded with Qumox cameras. A configuration of cameras was similar to previous user test scenarios: 12 cameras in a semicircle, full HD resolution, 30 fps.

The input data for 4D reconstruction was prepared in an equal manner as described in the previous chapters. The expectations for the final result were quite high, but it got stuck in the second step of the reconstruction process. The algorithm for reconstructing a dense cloud matched very few features between neighboring camera perspectives. A possible reason for this was found as individual frames were visually inspected and compared with one another. Videos from Qumox cameras are compressed and encoded, and consequently there is a noise-induced (micro-)pattern of compression artifacts within consecutive frames which is very inhomogeneous (see Figure 22).

Figure 22: Video compression and encoding artifacts

Unfortunately Qumox cameras don't provide an option to save uncompressed videos, thus we cannot confirm our hypothesis.

3.1.4 Benchmarking

The 4D Reconstruction pipeline will be part of the c-Space server. Windows 7 64 bit has been used for conducting the tests. For the calculations, a specific hardware setting is used to evaluate the performance of the implemented algorithms, as shown in Table 5 below:

	CPU	RAM	GPU
Hardware used	2 x Intel Xeon 3,1 GHz, 8 Cores	256 GB	GTX680 Phantom 4GB GDDR5

Table 5: hardware specification

The performance of the separate algorithms used in the implemented Multi-view Stereo approach was tested and is shown in the following table. All calculations are processed on the hardware regarding the table above.

	Number of images	Number of Aligned images	Alignment [min]	Build dense cloud [min]	Build mesh [min]	Build texture [min]	Whole 3D pipeline [min]
All photos	538	470	568	189	34	15	788
East video stream	259	259	450	166	5	11	634
Interior video stream	681	681	3269	769	6	33	4080
North video stream	376	376	1101	515	4	92	1714
West video streams	462	314	1972	242	6	15	2237
South video streams	468	159	1676	226	2	6	1910

Table 6: Performance of 3D Multi-View Stereo algorithms

In Table 6 there is a comparison among different 3D reconstruction test scenarios that were performed within this testing cycle. The third column represents the calculation time of the image alignment process which is the most time consuming process. In average it takes 80 % of the time of the whole reconstruction pipeline which is shown in the last column. The algorithm for building the dense cloud it the second time consuming process. In the fifth and sixth column the time needed to build a mesh with a mapped texture is presented. Regarding the complexity of the used algorithms the duration of the overall process is acceptable.

	Number of cameras	Number of Aligned cameras	Number of frames	Alignment [min]	Build dense cloud [min]	Build mesh [min]	Build texture [min]	Whole 4D pipeline [min]
Luxemburg – moving vehicle	12	11	19	10	7	9	14	40
Lab – Moose	12	12	23	9	16	17	39	89
Luxemburg – 4D product	12	10	79	28	40	21	29	118

Table 7: Performance of 4D Multi-View Stereo algorithms

In Table 7 the benchmarking results of the 4D test scenarios are summarized. In comparison to Table 6 the computation time of the algorithms are much lower. The reason for this is the lower resolution of videos and consequently the input images (frames). Additionally the alignment process has to be done only between 12 different camera positions. This leads to a much faster alignment step for each point in time (12 frames from each camera captured in the same point in time).

3.1.5 Quality checking

In order to check the quality of the outputs regarding the 3D & 4D reconstruction pipeline we compared the robustness of the alignment process in different scenarios and the complexity and completeness of the resulting 3D models. In Table 8 the results of the quality evaluation are shown:

	Number of images	Number of Aligned images	Percentage of alignment [%]	Number of faces in mesh
All photos	538	470	87	156 million
East video stream	259	259	100	13.3 million
Interior video stream	681	681	100	18 million
North video stream	376	376	100	9.7 million
West video streams	462	314	68	12.9 million
South video streams	468	159	34	3.7 million

Table 8: performance of Multi-View Stereo algorithms

The results demonstrate that even in difficult situations, like the deviations while the drone flight, the 3D reconstruction pipeline is generating qualitative results in an acceptable amount of time. Only two scenarios (West and South video stream) were not aligned completely (see Table 8) due to missing overlaps in images because of fast movements of the drone.

	Number of all cameras	Number of Aligned cameras	Percentage of alignment [%]	Number of faces in mesh (average)
Luxemburg – moving vehicle	12	11	92	95000
Lab – Moose	12	12	100	110000
Luxemburg – 4D product	12	10	83	86000

Table 9: performance of MVS algorithms

Results of 4D reconstruction pipeline are not so convenient. In the Genova test scenario, the 4D reconstruction was not performed at all, because it wasn't possible to synchronize frames (see Chapter **Error! Reference source not found.**). In the Luxemburg Moving Vehicle test scenario (see Chapter **Error! ference source not found.**) an additional action (hand-clap) in the beginning of a dynamic scene was performed in order to establish a good reference for synchronization of frames. The resulting 3D model was incomplete, because there were not enough 3D points reconstructed with the Multi-View Stereo algorithm. The problem occurred due to wet floor, which caused inhomogeneous pattern on the images captured from different camera locations. The following test scenario was performed in a controlled environment (see Chapter **Error! Reference source not found.**) and that improved the final output of the D reconstruction pipeline. As it can be seen in the chapter **Error! Reference source not found.**, another scenario was tested in similar settings and conditions, but the output 3D model was not convenient. After checking individual input images, an assumption was made: The video encoding and video compression have an extremely big negative influence for the reconstruction of a dense cloud. Further tests have to be conducted to evaluate the video compression on smartphones and tablets as well to achieve better results with the c-Space system.

3.1.6 Recommendations

The results obtained from the tests with static 3D reconstruction and dynamic 4D reconstruction in a controlled environment described in the previous sections indicate that a 4D reconstruction at promising quality will eventually be achievable.

However and more importantly for further development of c-Space, the tests revealed some new challenges that will be addressed in later stages of the prototype, as well as some useful information on

how to design the next testing environments to better address and benchmark the algorithms used in c-Space.

As already described in D2.1 for the Sorting/Alignment step within the 4D reconstruction module, some processing speed-up will be achieved by filtering out blurry images, since these and other images with similarly unusable content will not be considered in the computation, and additionally, their exclusion will make reconstruction more robust. Tests with videos for the 4D reconstruction experiments, but also tests with the frames extracted from the footage of the static 3D reconstruction experiment (Begato Fort) showed that there is some number of blurry frames within videos that can be filtered out.

An interesting and unexpected effect was revealed by tests with the 12 wide angle Qumox cameras chosen for use as testing setup: in contrast to other cameras, video compression was relatively strong, leading to significant changes between subsequent frames due to noise-induced compression artifacts. This effect drastically reduced the number of feature points necessary for 3D reconstruction. Hence, additional tests will be performed with different devices and compression settings, e.g. different smartphones, thereby also leading development more towards the real-world scenario c-Space will live in.

So far, image resolution per footage set was constant, but towards an algorithm robust to heterogeneous input, we will move towards tests with varying resolution within the same 3D / 4D reconstruction.

During the tests at Begato Fort, viewpoints changed for certain observers during capturing. This opened up the need of following each observer throughout the time sequence during the computation of single 3D reconstructions, in order to avoid jumping between perspectives when later viewing the final 4D reconstruction from a chosen perspective. Thus, tests will be designed where perspectives change deliberately during capturing.

During the 4D tests, e.g. the example of a moving car at Begato Fort, synchronization posed a hard challenge to frame sorting and assignment of coherent groups of frames corresponding to a certain point in time. This was due to the lack of timeframes, and as well due to the absence of a common synchronized clock. This shortcoming however will be solved once the components of c-Space grow together more tightly, and timestamps will be available per frame based on a synchronized clock.

Finally, the Luxembourg example underlined a very common problem with optical flow and feature detection as such: the number of identified feature points found in a scene shrinks significantly when the amount of texturing of objects is reduced. Due to this reason, the Luxembourg footage, showing a car on a homogeneously gray parking lot, in front of gray sky, lead to a critically low number of feature points found usable for 3D reconstruction. Another example is the floor in the Luxembourg indoor test scene, where the floor covers the largest part of the videos. Due to its reflective nature and repetitive, rather general structure, captured from different camera locations, floor regions captured in the camera frames hardly contributed to the feature set detected. As a consequence, further development will be extended on the search and more robust feature matching algorithms that will enable sufficient number of feature matching points especially in unstable environments.

3.2 CAI - System

3.2.1 Overview

The performance tests for the CAI system involve different load strategies and simulate the main cases of load over time for various methods. In particular three different tests were performed one for “scene status” returning the available media and its related details for a given scene (in our case the scene with sceneid=1 the only currently available scene in the database), the download test downloading one of the available medias in the database and finally the upload test involving the uploading of a new media.

Each test is divided into two parts. In the first one, “simple” strategy, runs a specified number of threads for testing each one of the three services. Each thread runs with a specified delay between each other, for instance running a functional test with 20 threads with 5 seconds delay with the delay being randomized by 1 to 5 seconds. With this strategy it is possible to assert the basic performance of the CAI, as well as to evaluate each method/service and discover if there are threading or resource locking issues. The second strategy is based on the “Fixed-Rate” strategy, a more complex strategy involving simultaneous use of the service by a number of threads (5 or 10 based on the service).

3.2.2 Benchmarking

The test environment concerning the client running the test is as follows:

CPU	Intel(R) Core(TM) i7-2670QM CPU @ 2.20GHz
Memory	8 Giga Bytes
OS	Linux kernel version 3.13.0-44
Connection	VDSL 50Mbps download 5Mbps Upload

Table 10: Test environment configuration

3.2.2.1 Access test using a “Simple” strategy

The parameters for the test include:

Strategy	Simple
Virtual users (threads)	10
Test Delay	1000
Random Threads	0.5
Running time	seconds 60

Table 11: Parameters of the test

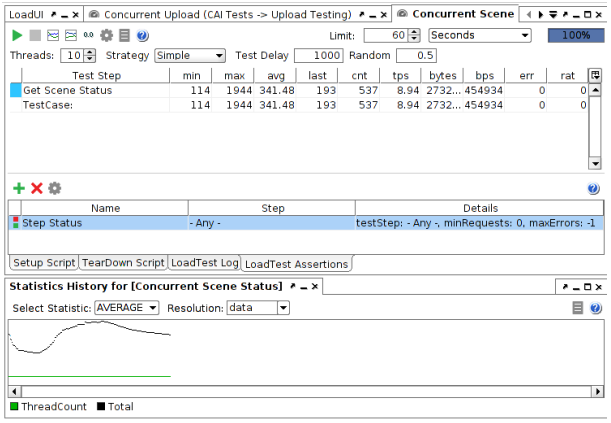

Figure 23: Test Interface – Simple test

Figure 23 depicts the test launcher interface for the given strategy and parameters.

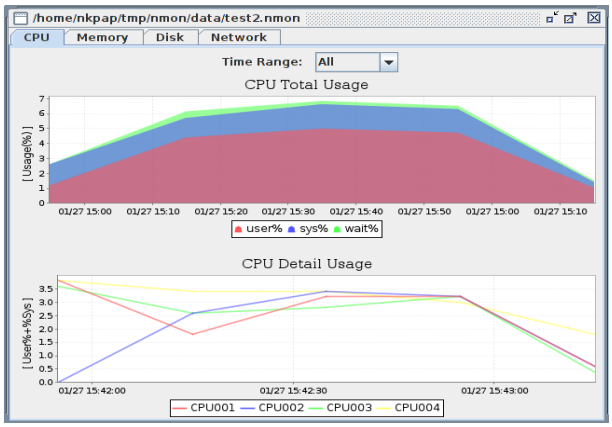

Figure 24: CPU usage – Simple test

Figure 24 depicts CPU usage for the Simple strategy using 10 concurrent threads. The load decreases as the first threads terminate.

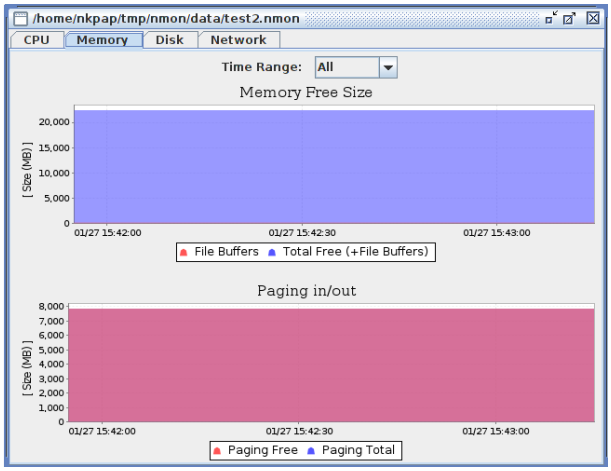

Figure 25: Memory usage – Simple test

Figure 25 depicts memory load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB out of 24GB were used.

Figure 26: Disk usage – Simple test

Figure 26 depicts disk I/O and read/write activity for the scene status test. Disk activity mainly concerns logging activities.

3.2.2.2 Access test using a "Fixed-Rate" strategy

The parameters for the test include:

Strategy	Fixed-Rate
Virtual users (threads)	10
Test cases per second	10
Max number of threads	100
Running time	60 seconds

Table 12: Parameters of the test

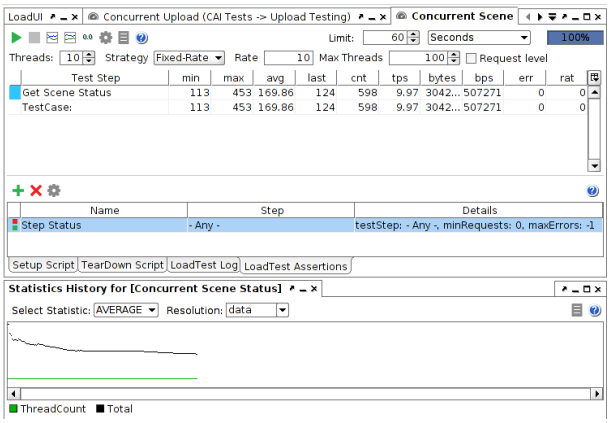

Figure 27: Test Interface - Fixed-Rate test

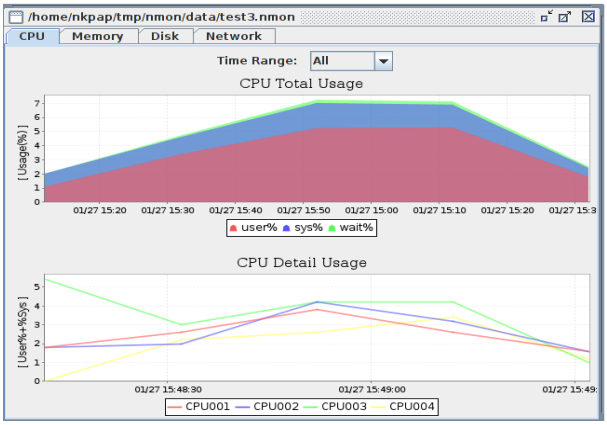

Figure 28: CPU usage - Fixed-Rate test

Figure 27 depicts the test launcher interface for the given strategy and parameters.

Figure 28 depicts CPU usage for the Fixed-Rate strategy using 10 concurrent threads. The load decreases as the first threads terminate.

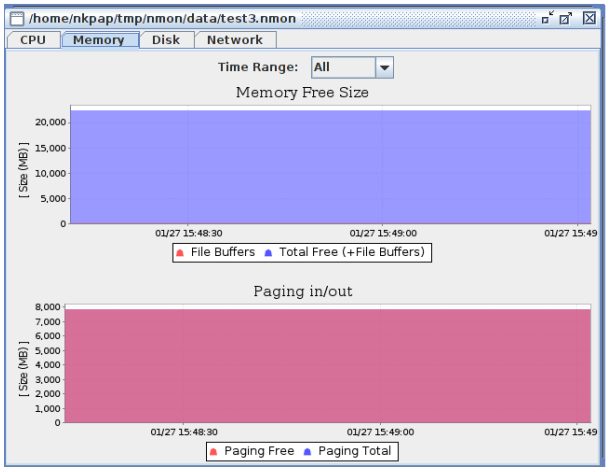

Figure 29: Memory usage - Fixed-Rate test

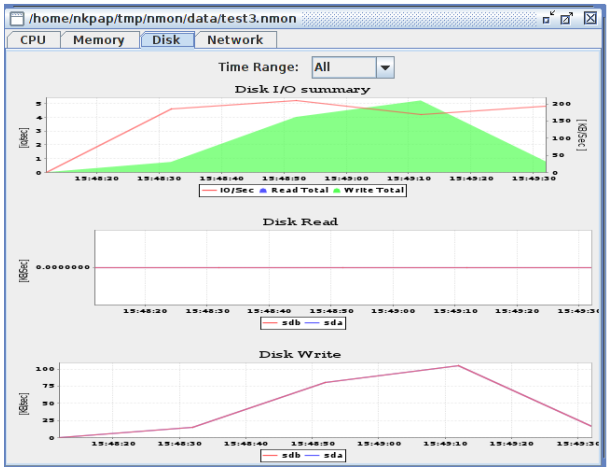

Figure 30: Disk usage - Fixed-Rate test

Figure 29 depicts memory load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB out of 24GB were used.

Figure 30 depicts disk I/O and read/write activity for the scene status test. Disk activity mainly concerns logging activities.

3.2.2.3 Download test using a "Simple" strategy

The parameters for the test include:

Strategy	Simple
Virtual users (threads)	5
Test Delay	1000
Random Threads	0.5
Runs per Thread	1

Table 13: Parameters of the test

The virtual users have been reduced to 5 because. A higher number of Threads resulted in java.lang.OutOfMemoryError: Java heap space. This limitation concerns the client side. Although the client is a "powerful" desktop system (as described in section 7.2) it fails when used for a higher number of concurrent threads.

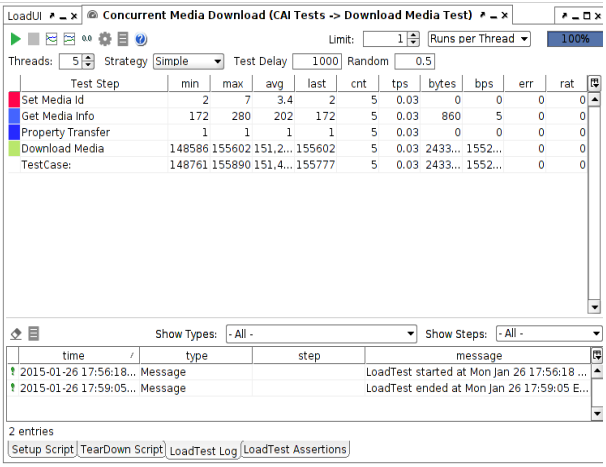

Figure 31: Test Interface - Simple test

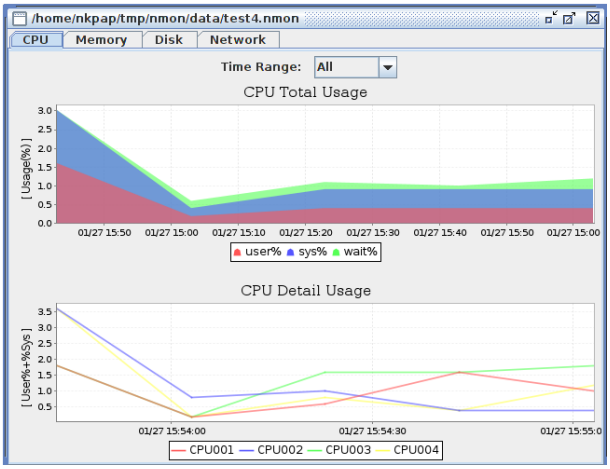

Figure 32: CPU usage - Simple test

Figure 31 depicts the test launcher interface for the given strategy and parameters.

Figure 32 depicts CPU usage for the simple strategy. The four cores present a similar behaviour during the test.

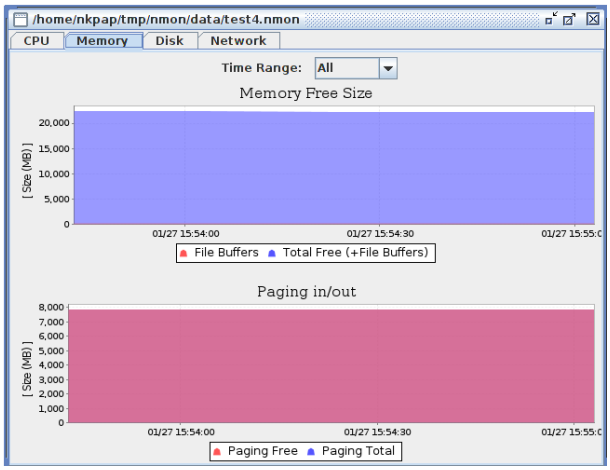

Figure 33: Memory usage - Simple test

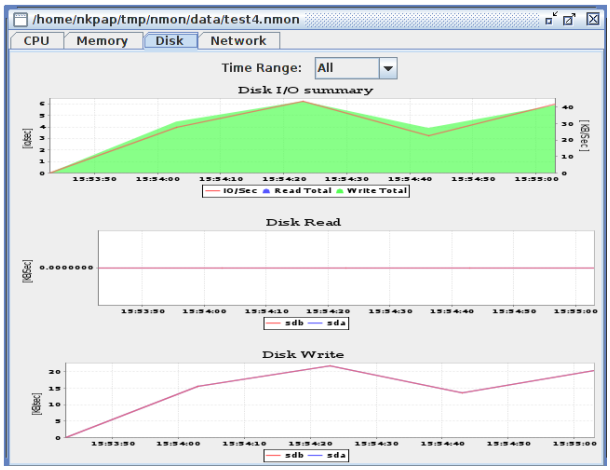

Figure 34: Disk usage - Simple test

Figure 33 depicts memory load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB out of 24GB were used.

Figure 34 depicts disk I/O and read/write activity for the Download test. Disk activity mainly concerns media retrieval from the File system as well as logging activities

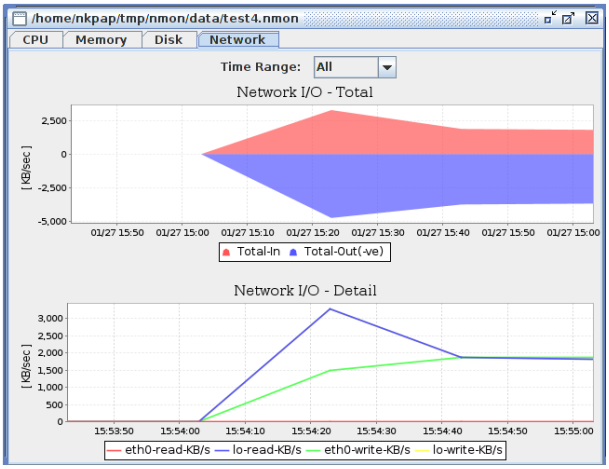

Figure 35: Network Traffic - Simple test

Figure 35 depicts the network traffic for the download test. Monitoring was started before the actual stress test. This explains the initial inactivity in terms of network I/O.

3.2.2.4 Download test using a "Fixed-Rate" strategy.

The parameters for the test include:

Strategy	Fixed-Rate
Virtual users (threads)	5
Rate	1
Max Threads	5
Runs per Thread	1

Table 14: Parameters of the test

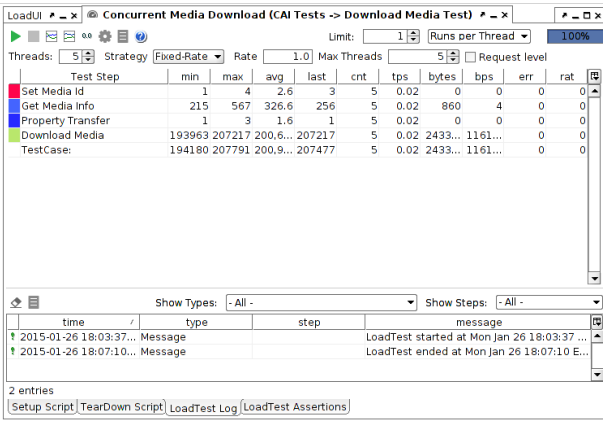

Figure 36: Test Interface - Fixed-Rate test

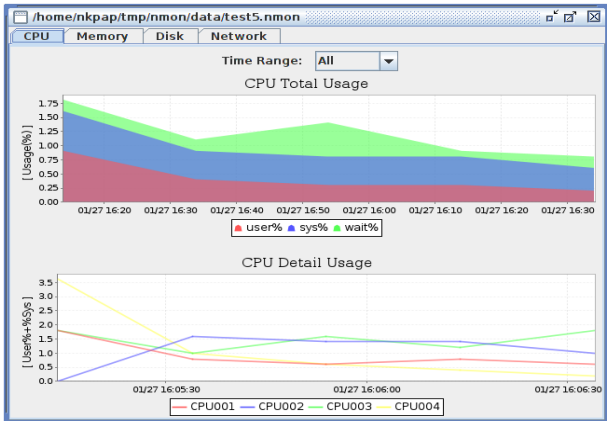

Figure 37: CPU usage - Fixed-Rate test

Figure 36 depicts the test launcher interface for the given strategy and parameters.

Figure 37 depicts CPU usage for the Fixed-Rate strategy. The four CPU cores present a similar behaviour during the test.

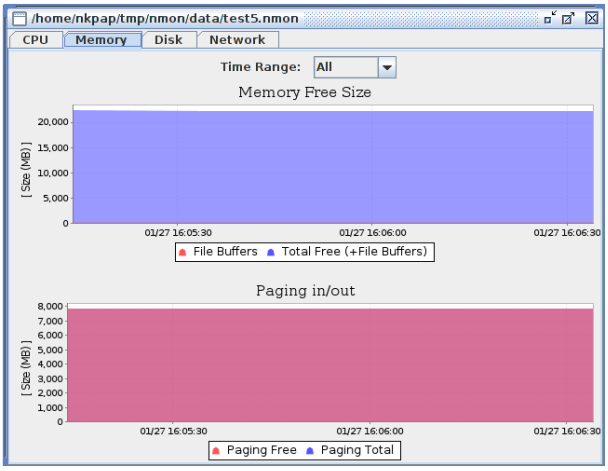

Figure 38: Memory usage - Fixed-Rate test

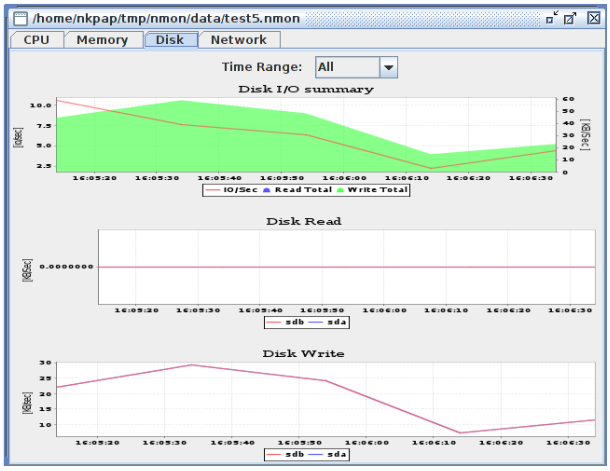

Figure 39: Disk usage - Fixed-Rate test

Figure 38 depicts memory load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB out of 24GB were used.

Figure 39 depicts disk I/O and read/write activity for the Download test. Disk activity mainly concerns media retrieval from the File system as well as logging activities.

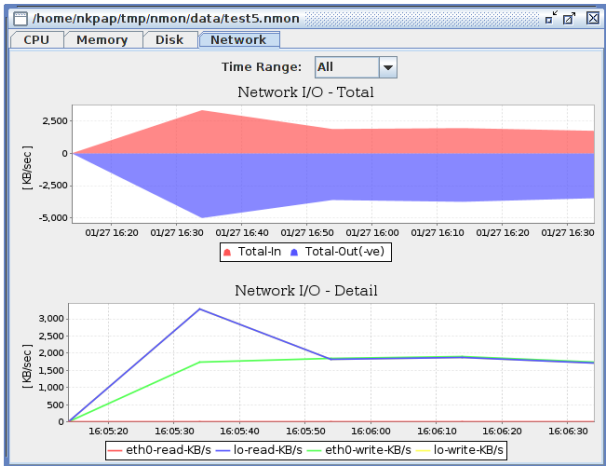

Figure 40: Network Traffic – Fixed-Rate test

Figure 40 depicts the network traffic for the download test. Monitoring was started before the actual stress test. This explains the initial inactivity in terms of network I/O.

3.2.2.5 Upload test using a "Simple" strategy

The test includes three distinct steps and involves a Groovy script to upload a media file to the CAI server. It renames using the current timestamp in order to have different filenames for the uploaded files. The Groovy script is shown in the following.

```
// get request
def request = testRunner.testCase.getTestStepByName( "Upload video" ).testRequest
// clear existing attachments
for( a in request.attachments ) {
request.removeAttachment( a )
}
def groovyUtils = new com.eviware.soapui.support.GroovyUtils( context )
// get file to attach
def file = new File("/home/nkpap/tmp/CAI/TESTs/20141231_154952.mp4" )
if ( file == null) {
log.error "bad filename"
throw new FileNotFoundException()
}
else
{
Thread.sleep(Math.abs(new Random().nextInt() % 500))
// attach and set properties
def attachment = request.attachFile( file, true )
//attachment.contentType = "application/octet-stream"
//attachment.setPart( "" )
attachment.setContentID("file")
attachment.setName("test_video_"+new Date().getTime()+".mp4")
log.info(attachment.getName())
}
```

The parameters for the test include:

Startegy	Fixed-Rate
Virtual users (threads)	5
Rate	1
Max Threads	5
Runs per Thread	1

Table 15: Parameters of the test

The virtual users have been reduced to 5 because. A higher number of Threads resulted in java.lang.OutOfMemoryError: Java heap space. This limitation concerns the client side.

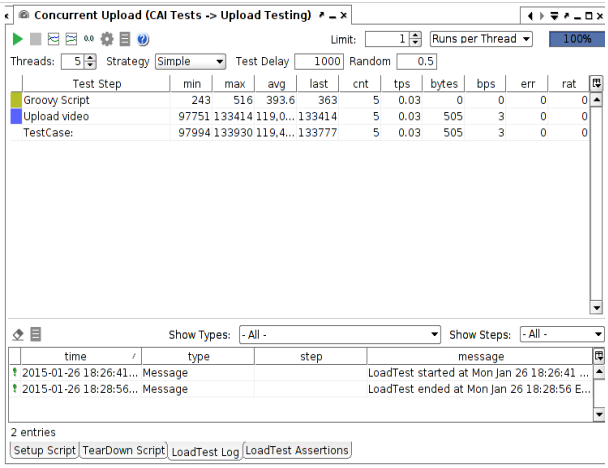

Figure 41: Test Interface - Simple test

Figure 41 depicts the test launcher interface for the given strategy and parameters.

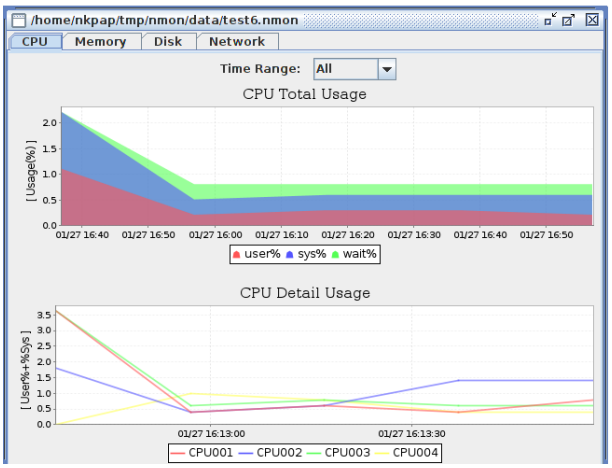

Figure 42: CPU usage - Simple test

Figure 42 depicts CPU usage for the Simple strategy. The four CPU cores present a similar behaviour during the test.

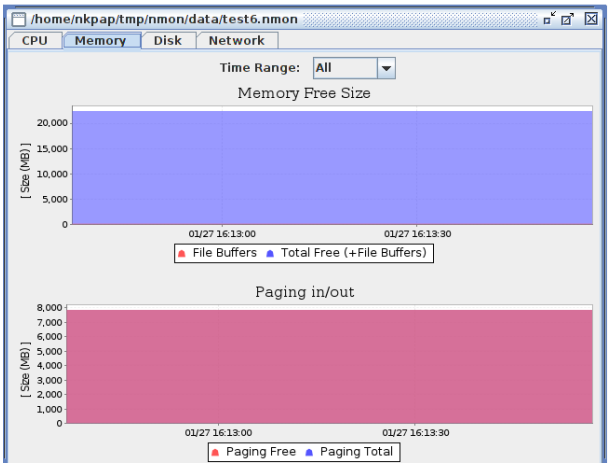

Figure 43: Memory usage - Simple test

Figure 43 depicts memory load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB

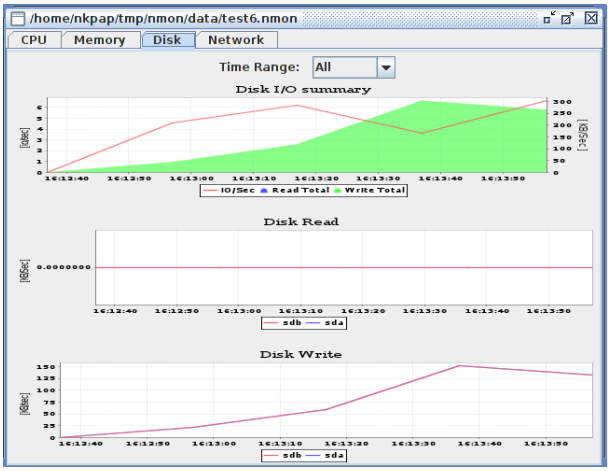

Figure 44: Disk usage - Simple test

Figure 44 depicts disk I/O summary for the Upload media Test. Write increases over time as more threads try to upload.

out of 24GB were used.

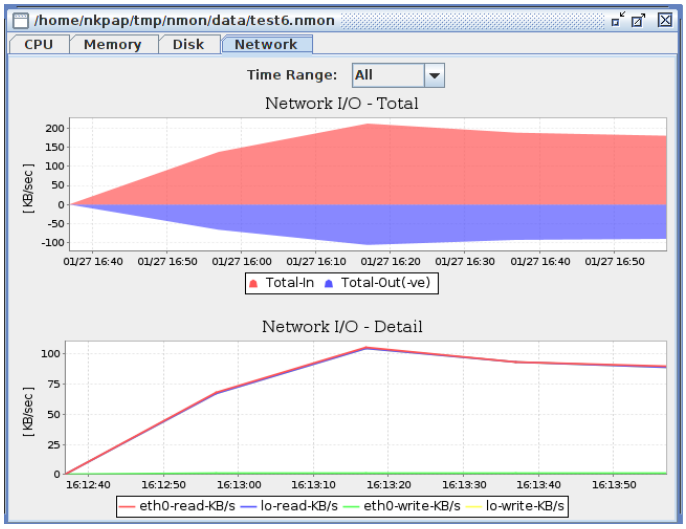

Figure 45: Network Traffic - Simple test

Figure 45 depicts the network traffic for the download media test.

3.2.2.6 Upload test using a "Fixed-Rate" strategy.

The parameters for the test include:

Strategy	Fixed-Rate
Virtual users (threads)	5
Rate	1
Max Threads	5
Runs per Thread	1

Table 16: Parameters of the test

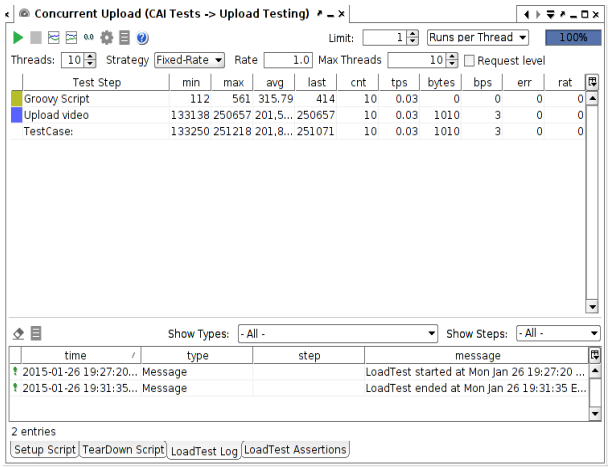

Figure 46: Test Interface - Fixed-Rate test

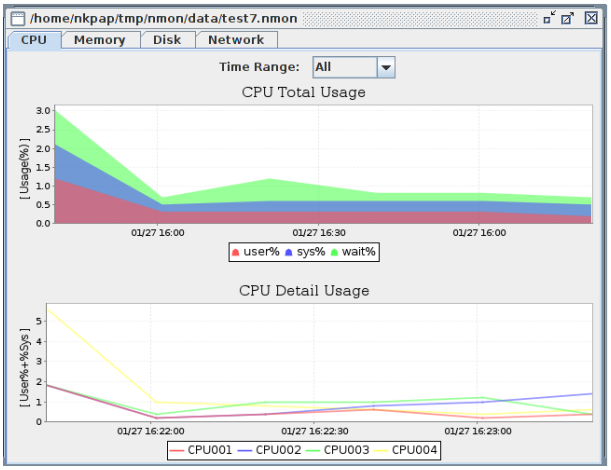

Figure 47: CPU usage - Fixed-Rate test

Figure 46 depicts the test launcher interface for the given strategy and parameters.

Figure 47 depicts CPU usage for the Fixed-Rate strategy. The four CPU cores present a similar behaviour during the test.

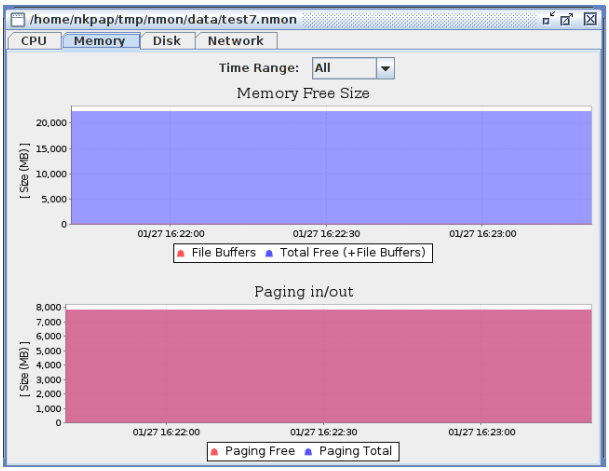

Figure 48: Memory usage - Fixed-Rate test

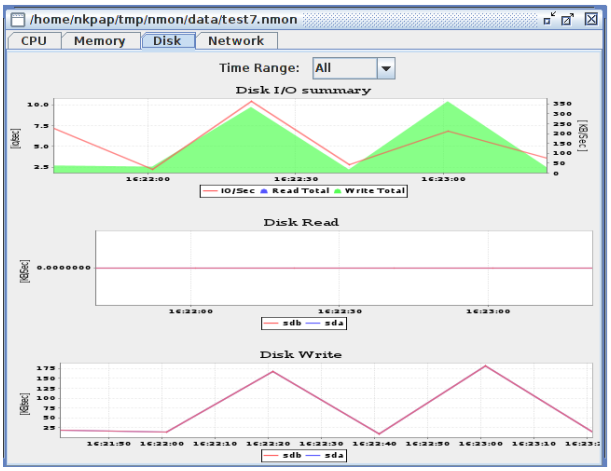

Figure 49: Disk usage - Fixed-Rate test

Figure 48 depicts load during the test. The majority of the total memory of the machine remains underutilized as the test runs. Only 2GB out of 24GB were used.

Figure 49 depicts disk I/O summary for the Upload media Test. Write increases over time as more threads try to download.

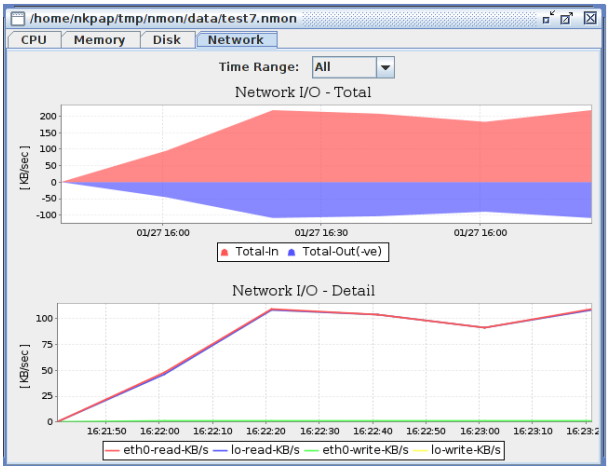

Figure 50: Network Traffic – Fixed-Rate test

Figure 50 depicts the network traffic for the upload media test.

3.2.3 Quality checking

The first two tests are read-only as far as the database is concerned (no new records are written into the database or the remote file system), while the third test creates rows in the media table of the CAI as well as the remote file system (for each uploaded file a directory is created and named after the prefix "Media_" and the media id obtained in the database). The media file is then put in the media directory. The quality of the tests involves also manual checking, for the first two tests the result is compared with the content into the CAI system using the hashing algorithm SHA256: the downloaded file is compared with the "original" content. The quality check reported no corruptions or divergences from the "correct" values.

For the third test for each process of the test, the uploaded file has been compared with the original file using the hashing algorithm SHA256. All files were integral and no corruption has been reported.

3.2.4 Testing tools

The testing tools used comprise the soapUI test suite for modelling and performing the various load strategies and the NMON application to log performance on the server side and provide load graphs for the CPU, memory, disk I/o and network.

3.2.5 Recommendations

All tests to the CAI system have been conducted with success. The CAI system passed the tests with low to medium load although some tests saturated the client performing the tests.

3.3 Affective module

3.3.1 Overview

Current state of the Affective module performs user's emotional recognition based on the facial expression. The module has been evaluated in terms of performance and quality, as well as validating the exposed API for intercommunication with other modules.

In summary, a great performance has been achieved allowing the module to be used in targeted mobile devices, and a good quality has been reached in the facial recognition under different conditions. However, limitations on the allowed rotation and the loss of tracking under rapid movements limit the robustness of the system. Hence, a different approach that could overcome these issues is being pursued.

3.3.2 Benchmarking

Since one of the main objectives of the Affective module is to perform facial affect recognition on mobile devices, its performance has been evaluated on iOS and Android devices. From the results presented in Table 17 it can be seen that the average computation time needed for the initialization process and the following tracking process are low enough to perform a real-time recognition.

Device	Initialization (ms)	Frame tracking (ms)
iPad 2	60	42
LG Nexus 5	52	26

Table 17: Affective module average computation times

3.3.3 Quality checking

In order the evaluate the performance of the recognition system in different configurations of viewing angle and illumination, the CMU Pose, Illumination, and Expression (PIE) database[4] is used. The database contains 41.368 images of 68 people under 13 different poses, 43 different illumination conditions and 4 different expressions. In Figure 51 some examples of the database used in the evaluation are shown, with the adjusted 3D face model.

[4] Sim, T., Baker, S., & Bsat, M. (2003). The CMU Pose, Illumination, and Expression (PIE) database. In Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition (pp. 53–58). IEEE. doi:10.1109/AFGR.2002.1004130

Figure 51: Examples of facial tracking in the CMU PIE database

The followed approach, the combination of feature detection and the deformable backprojection (FFBP) described in D3.6, is compared to another three alternatives: Holistic Approach (HA), Constrained Local Model (CLM) and Supervised Descent (SDM). Fitting errors for the four approaches using the presented 3D face model are shown in Table 18. Obtained results have less error than HA and similar values to CLM. However, the followed approach does not depend on the quality of the trained model.

	Fitting error	
	Mean	**Stdev**
FFBP	14.93	7.35
HA	32.42	17.16
CLM	13.44	7.82
SDM	9.05	4.14

Table 18: Fitting errors comparison in the CMU PIE database

However, during the evaluation it was found that there is a limitation on the out-of-plane rotations (-30 deg, 30 deg) and that when rapid movements occur, tracking degrades and a fitting reinitialization is needed. Automatic recovery from this situation was explored, but a different approach has been proposed that would also improve overall recognition.

3.3.4 Testing tools

A sample Android mobile application was developed to evaluate the support on different devices. Using the testing platform TestFairy[5], it was tested without any crash detected on 20 unique devices of the following models:

- Xiaomi - MI 4W
- Huawei - Ascend G510

[5] http://testfairy.com/

- JYT - JY-G4C
- Sony - C6903
- Sony - C1905
- LGE - Nexus 5
- Motorola - XT1032
- Motorola - moto G
- NVIDIA - TegraP1640
- Samsung - Galaxy S2
- Samsung - Galaxy S3
- Samsung - Galaxy S5
- Samsung - Galaxy Note 3
- Samsung - Galaxy Tab 2 10.1

3.3.5 Software validation process

At software level, Affective module API, shown in Figure 52, has been validated by performing unit testing. Using the JUnit[6] framework, each API method has been tested with different combinations of input configurations.

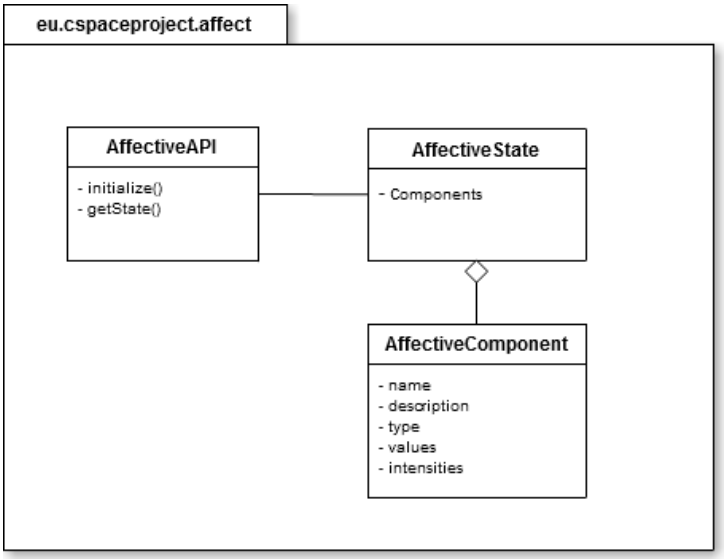

Figure 52: Affective API

3.3.6 Recommendation

In order to increase the reliability and robustness of the facial emotional measurement, a different approach to detect the facial expression has been suggested, which performs simultaneously the facial tracking and the expression recognition processes. Although dominant approach on 3D recognition of

[6] http://junit.org/

facial expressions is based on computing a time-varying description of facial actions and then to estimate from them the facial expression, both processes are interrelated and performing both simultaneously can allow to perform facial expression recognition when partial occlusions, rapid movements or video discontinuities are present.

3.4 Geotagging

3.4.1 Overview

This software module is used to tag a video stream or image with location information as well as informative metadata provided either automatically through a repository such as Wikipedia or from the users themselves. The module is also responsible for uploading the data to the c-Space repository. The geotagging module consists of two parts:

- Getting the GPS position, which takes place on the client-side (Android application)
- Obtaining location information from third party sources (DBPedia[7]), which takes place on the CAI.

3.4.2 Benchmarking

The time required to fetch DBPedia data on the CAI is almost instantaneous, since it only requires a call to the DBPedia service, so it can be safely considered negligible. The CAI handles all network errors. If the DBPedia service cannot be contacted, then no geotagging info is added to the media content.

Almost all of the elements of server's controlled test set were documented in Wikipedia and as a result the application returned relevant tags (up to 95%).

In the case of GPS tracking, if no GPS provider is present then location tracking cannot take place. In addition, in the case of the EGNOS-SISNeT combination, a network (3G/4G/Wi-Fi) connection is required. It is up to the client application to handle tracking errors.

The time required for obtaining the GPS position depends on the Android device GPS receiver used. During the tests conducted using the EGNOS[8] hardware receiver, getting an initial EGNOS fix requires at least 10 seconds, since the SISNeT[9] has to be contacted and the EGNOS SW receiver has to calculate the initial position. After that, location updates take place on average every 2-3 seconds.

3.4.3 Quality checking

Not all the tags were relevant to the particular scene. The tags retrieved depend on the available documents present for the particular source (Wikipedia) and it is not a limitation of the application.

[7] http://dbpedia.org/About
[8] http://egnos-portal.gsa.europa.eu/
[9] http://www.egnos-pro.esa.int/sisnet/index.html

3.4.4 Recommendation

In the case of position tracking, once and if the Galileo service is deployed and made publicly available by GSA, it could replace/enhance the EGNOS module, as an alternative tracking provider.

When trying to retrieve DBPedia data, a scheduler could be created which would periodically check all media content not containing geo-tags and re-try to contact the DBPedia service.

More sources can be included in the retrieval process to maximize discovered tags and information (i.e. Twitter, Facebook).

3.5 Video Data Extraction Software Module

3.5.1 Overview

This software module is used to extract images and video streams from the users of c-space and upload them to the server for processing and fitting onto the 3D Models.

The tests were not performed on 3G/4G networks since this requires a user base equipped with mobile applications and phones to be present. All tests were conducted using Wi-Fi on VDSL[10] and ADSL[11] connections with ordinary desktop machines and the findings revealed that the traffic generated from our test machines were unable to saturate the broadband connection as of the CAI server (symmetric 100Mbps with SLAs etc.).

3.5.2 Benchmarking

Upload resume functionality has been implemented both on the client (Android) as well as the server (CAI) sides. Interrupting a video upload by disabling the network does not corrupt the uploaded video data on the server, simply leaving a partial/temporary upload on the CAI. When and if the user retries to upload the video, the upload is picked up from where it was interrupting. Only when the video is completely uploaded to the CAI does it gets saved and indexed there.

Further to this, multiple upload has been tested with two different test strategies using a number of concurrent threads (see deliverable D3.2, section relevant to the upload test).

3.5.3 Quality checking

Video tracking data heavily depends on the following factors:

- GPS receiver precision
- Timestamp precision

[10] http://en.wikipedia.org/wiki/Very-high-bit-rate_digital_subscriber_line
[11] http://en.wikipedia.org/wiki/Asymmetric_digital_subscriber_line

Tests conducted with 12 sample videos provided by partner FRAUNHOFER, along with one GPS position (calculated by Google maps) and time-stamp (manually measured) per video, caused no problems when reconstructing the 3D models. This strongly indicates that position provided by GPS hardware will be adequate, compared to a location picked manually from Google Maps.

As far as temporal accuracy is concerned, once "real-world" videos (along with richer tracking data) are captured by the mobile application and fed to the 4D reconstruction pipeline, more detailed conclusions will be reached. As described in D.3.4, it is emphasized that the time offset between the Android device and the NTP[12] time server must be measured as close as possible (in millisecond accuracy) to the start of video recording.

If no video tracking data is included, a video upload is not rejected. Tracking data extraction simply does not take place, however the video is stored and indexed normally.

3.5.4 Recommendation

Adding video tracking data "offline" (i.e. after a video has been uploaded to the CAI) could be implemented, in order to facilitate testing purposes.

4 Final recommendation

4.1 Overall results as integrated system

The validation of the integrated mobile client was based on the following evaluation instruments: c-Space Instrument for Measuring Creativity, JUnity testing unities, and Mobile Client Evaluation Checklist. The results of the mobile client evaluation checklist were positive. The mobile application was successfully installed on the mobile device and it was easily recognizable by the user. Furthermore, the application was uninstalled without errors. Re-installations worked as expected. More network checks have to be implemented to make the application more robust to network issues. We have detected minor errors when the network goes down, which should have returned messages like "Network unavailable. Please try again later". The items regarding voice calls, SMS and memory handling didn't reveal any unexpected surprises. Soft and physical buttons presented the behavior that was defined or expected. The application sensors behaved according to design specifications. No user's physical disabilities were considered for the first integration phase, but mechanisms like voice control are expected in future releases. The measure of the performance was conducted through benchmark software/services specifically designed to measure GPU performances and that works in emulator too. For more information about the performance of the mobile client and known issues see previous sections.

[12] http://en.wikipedia.org/wiki/Network_Time_Protocol

4.1.1 Experimental modules also integrated

In addition to the results of deployed modules, an experimental version of immersive game-based localized social media interactions model was also tested. Users can interact with spatially located 3D content that crystalizes on Big Data User-centric model that is particularly concerned with value creation for users. Departing from the Big Data User-centric model for social media platforms, we conceptualized value as a meaningful interest-based content discovery through interaction with other users, content, and the system. Interactivity conditions consists of noninteractive state, when new messages are not related to previous messages; reactive, when new messages are related only to a specific, often times prompted message; and interactive, when new messages can be related to a number of previous messages having a specific relationship between them. Interactive interest-based environment thus aimed to increase degrees of adaptability, user agency, choices, and flexibility leading to some of the positive outcomes of Interactivity such as user satisfaction. Below, we have a representation of UGC with a spatial component.

Figure 53: 3D reconstruction of Trento

To implement our mobile-oriented framework geared towards user-centric experience, we considered the following six dimensions of perceived interactivity proposed by McMillan[13]. Direction of communication – the ability to return messages, in addition to one-way communication; time flexibility – the ways to engage in communication in real time as well as being able to retrieve archival conversations; sense of place – creation of a common virtual communicative context; level of control – user agency to select responses in the most appropriate context, ability to track messages and identify if they were delivered; responsiveness – relates to the concept of sequential interrelation between messages.

[13] S. J. McMillan, "A four-part model of cyber-interactivity some cyber- places are more interactive than others," New Media & Society, vol. 4, no. 2, pp. 271–291, 2002.

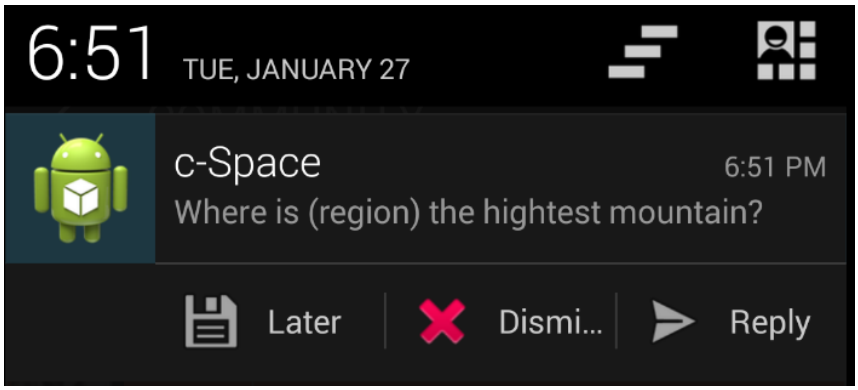

Figure 54: Interaction system

We performed also initial tests to the mobile streaming framework, not yet fully integrated. The current streaming framework takes into account the client's rendering capabilities, limitations in the network bandwidth and user actions to automatically adapt the resolution of the 3D model. Additionally, it optimizes the set of memory operations that have to be performed at the client side, so the user experience while visualizing the 4D content is as fluid and jitter free. Figure 55 provides an overview of current results.

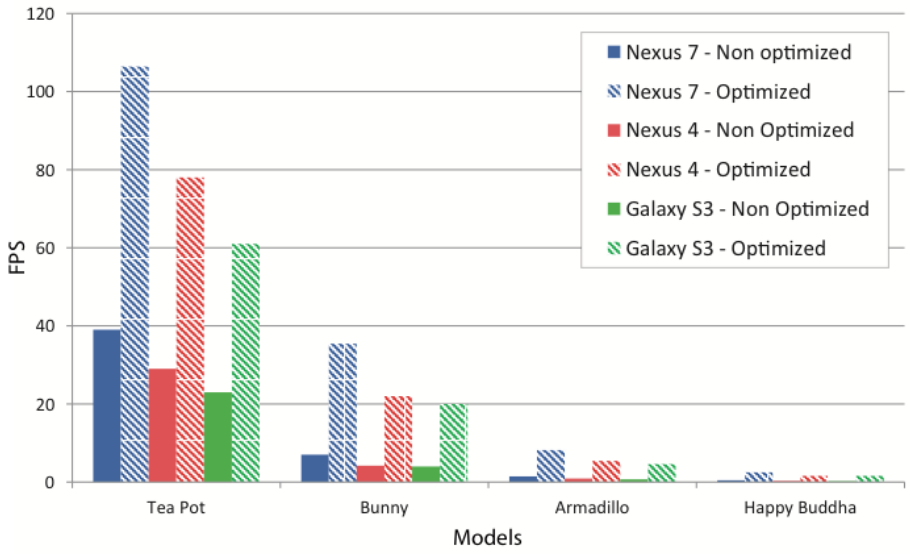

Figure 55: Performance comparison of optimized and non-optimized streams.

Lastly, the questioner that attempts to evaluate the impact of c-Space tool in the user creativity was not conclusive since both user interaction and recommendation modules were not fully integrated. Hence, the user experience was limited, e.g. mechanisms for opportunity and recognition and collaboration could not be assessed.

Regarding the server-side of the c-Space system, tests with an integrated system have to be conducted in the next testing cycle. This includes tests with videos and images captured by users using the c-Space app. Additionally the captured videos and images have to be enriched with a common time shared by the

CAI Web service to all running c-Space apps. This allows the 4D reconstruction pipeline to synchronize all the media footage captured within one scene in order to get a robust reconstruction.

4.2 Use of issue tracker

GitLab is an open source software to collaborate on code as well an issue tracker. The c-Space partners have been using it to manage the mobile application client and to keep track of open issues. The public URL is http://gitlab.c-spaceproject.eu. The use of this issue tracker has been limited because most modules were just recently uploaded. In the near future, and as test continue, we should see more engagement in this section as the integration becomes more complex.

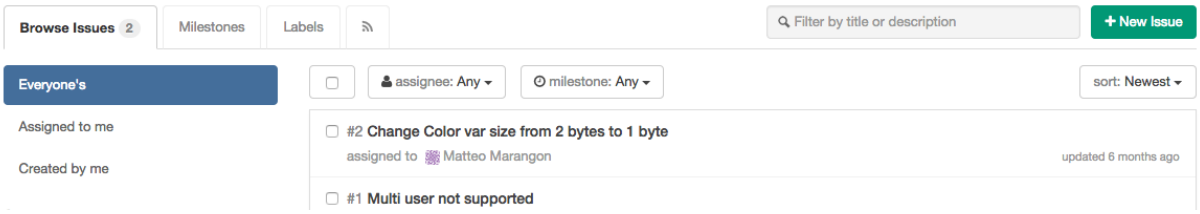

Figure 56 – GitLab issue tracker

4.3 Next steps and actions

The most frequently reported limitation of the c-Space augmented reality application (part of the streaming framework), in its current state of development, is the user cognitive overload. It becomes complicated for users to visualize in simultaneous multiple 4D scenes that share the same space. Current mechanisms allow users to select the model they want to see but further improvements should allow the visualization of multiple scenes when possible. Hence, it requires a topological understanding of each scene. We were aware of this issue however the implementation didn't make it into the first integration. A different approach to detect the facial expression is been studied, which performs simultaneously the facial tracking and the expression recognition processes. The geotagging module will be extended with additional data sources to provide more accurate tags. Focus will be given to temporal events. Regarding the Video Data Extraction Module video tracking data could be provided offline, to support testing cycles in the future.

The results obtained from the tests with static 3D reconstruction and dynamic 4D reconstruction in a controlled environment described in the previous sections indicate that a 4D reconstruction at promising quality will eventually be achievable.

Finally, a very common problem with optical flow and feature detection as such: the number of identified feature points found in a scene shrinks significantly when the amount of texturing of objects is reduced. Due to this reason lead to a critically low number of feature points found usable for 3D reconstruction. As a consequence, further development will be extended on the research and more robust feature matching algorithms that will enable sufficient number of feature matching points especially in unstable environments.