Intelligent Picture Browser
-
Given One
-
NGUI report

Pierre Gerard - Matteo Marra - Bruno Rocha Pereira

December 19, 2016

1 Introduction

New generation user interfaces are developing day by day, with new tools, devices and different use cases showing up.

Among those devices, Virtual Reality sets are spreading as they get more affordable and many research teams start working on it.

Our brainstorming led us to imagine a future where we could use Virtual Reality on a common basis. We thought about what an user would like to have which doesn't exist yet, and how to implement it to make the user feel comfortable in this totally new environment.

The system will offer the user a collection of pictures in a Virtual Reality environment, allowing him to browse them in a smart way.

2 Problem to be solved

Since the numeric revolution, humans tend to take a ton of pictures. It is especially true during their holidays, usually coming back with thousands

of pictures. The problem that arises then is that humans don't usually have a mean to explore them all other than browsing through them one by one. Therefore, the idea for the project is to create a *Intelligent Picture Browser (IPB)* to fill that need.

3 Requirement Analysis

The product should browse the user's complete gallery of pictures and select a subset of them. The subset selection will be based on a keyword/tag which is automatically bound to pictures by an AI algorithm.

The user will be able to interact with the interface through the virtual reality set and be able to select tag and some settings via voice.

3.1 Data requirements

The interface should work with any set of pictures in a *.jpg* format. However, for testing and demonstration purpose, a pre-loaded set of 300 holiday pictures will be available.

3.2 User characteristics

The interface will be design to fit almost every individual having the capacity to use a virtual reality set and voice recognition. It will target technological novice as well as professional. Unfortunately, people with disabilities preventing them from using a VR set or voice recognition won't be able to use our system.

3.3 Usability goals

The usability of this project is going to be as straightforward as possible and will not require any particular skill or educational background. Nonetheless, having a regular access to technologies and computers will bring smoother use. In brief, it should be easy to learn, efficient and effective.

3.4 User experience goals

The main goal is to make the user experience the moment immortalized by the pictures and allow him to feel the same emotions once again.

4 Design and prototyping

Concerning the design, implementation and validation of the next generation user interface, we worked in an iterative way. The idea was to create multiple evolutionary prototypes. Each prototype was then presented to the class.

Thanks to that method, we were able to create a final product on time and validated its usability.

4.1 Iteration 1 : Problem defined

The first iteration consisted mainly in sitting in a comfortable room and brainstorming about which next-generation interface we would design. Many ideas had emerged and the one that convinced us the most was to innovate in the field of picture browsing. Indeed, multiple technologies have recently appears on the market and we though they could be use to enhance usability and user experience of such systems. The first and main assumption made on behalf of the user is that the user possesses a lot of picture but doesn't have the time to browse or sort them all, which is what we want to solve. The added value of our idea compared to existing picture browsers was an intelligence which recognizes effortlessly the content of the user's pictures and an immersive way to browse through them thanks to new technologies: virtual reality and voice recognition.

4.2 Iteration 2 : Requirements defined

The second iteration was about planning the realization of the idea for the interface and getting the requirements right. We presented the requirements in the section 3. The planning consisted of a Gantt chart with a general idea of the timeline we were going to follow for the next 3 months. Of course, due to unpredictable challenges some parts of it got delayed and some of them were realized quicker than expected.

4.3 Iteration 3 : Low-fidelity prototype

This iteration was all about making sure the product was feasible and that the requirements could be reached by making a low-fidelity prototype with all components working separately.

The idea here was to find solution for each of those components :

- A Neural Network capable of automatically tagging the images,
- A framework to support the HTV Vive Virtual reality set,
- A speech recognizer compatible with the chosen VR framework,

More information about the technologies used can be found in the architecture section 5.3 .

4.4 Iteration 4 : High fidelity prototype

The goal of this iteration was to manage to build a fully-working high-fidelity prototype including the main features of the interface. The prototype

built at this point was capable of handling voice-recognized keywords as well as showing the corresponding images to the user in the Virtual Reality environment.

4.5 Iteration 5 : Prototype improvement

This iteration focused on user goals and on usability through improvement of the first high-fidelity prototype. Following users feedback, we changed the way the pictures were displayed : instead of showing a "wall" of pictures we decided to surround the user with picture, circling the user with pictures. We also added more advanced queries, where a user can do the union and the intersection of subsets of pictures using combination tags.

4.6 Iteration 6 : Final product

This iteration was about finishing and polishing our user interface. We first added visibility to the features implemented and that a user can possibly use. To do that we added to tag suggestion and selected tag to the screen.

We also added feedback to the user. When he looks a pictures it gets slightly bigger making him aware of the currently selected .

We also mapped buttons the Vive controller to the up and down movements in the virtual world. We also added the possibility to move the image closer or further by saying the corresponding keywords.

Finally, we also made the evaluation of the interface presented in section 6.

5 Technical : Intelligent Picture Browser

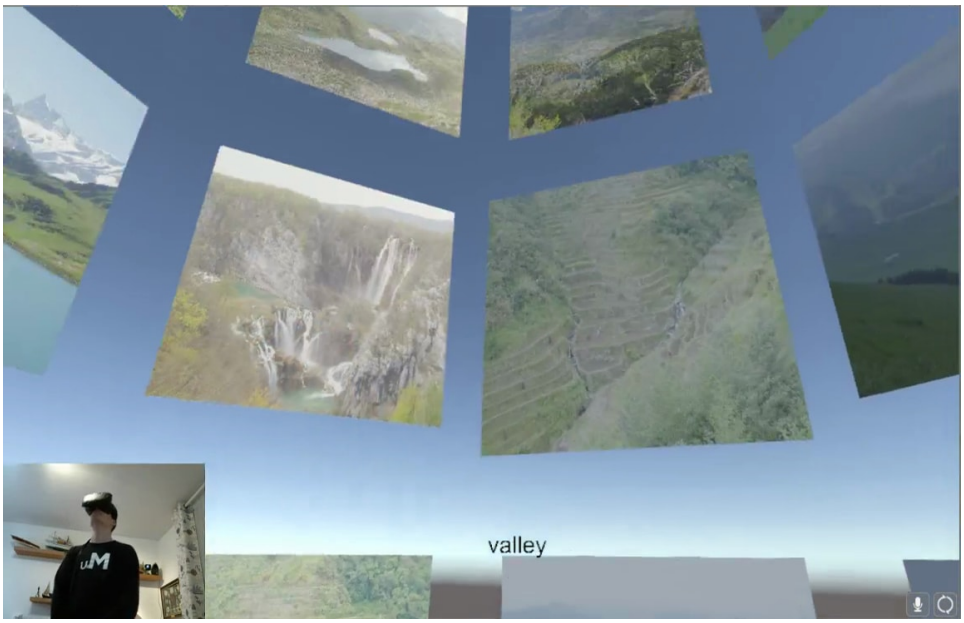
The Intelligent Picture Browser is built to work on state of the art technologies, in order to offer to the user a new kind of interaction. It uses Virtual Reality for displaying and partially interacting, a neural network to categorize the pictures and speech recognition for selecting them.

It has a multimodal interface, that involves visualization, speaking and movement. Other than interacting with the speech recognition, some commands are linked to the physical controllers of the HTC vive, that permits, in those cases, to have faster interaction.

5.1 Functionalities

Functionalities are divided in two sections: basic and advanced. The first one represent the basics of the application, the second ones are less intuitive operations that were added to complete the user experience. After executing the tagging script, images are flagged with one or more tags, that are considered the most probable for them.

5.1.1 Basic functionalities



The user can:

- Select pictures using one of their tag
- Navigate through the picture shown around him
- Select a picture to see it bigger
- Look at suggestions of tags.
- Open an helper that will explain how to activate the commands

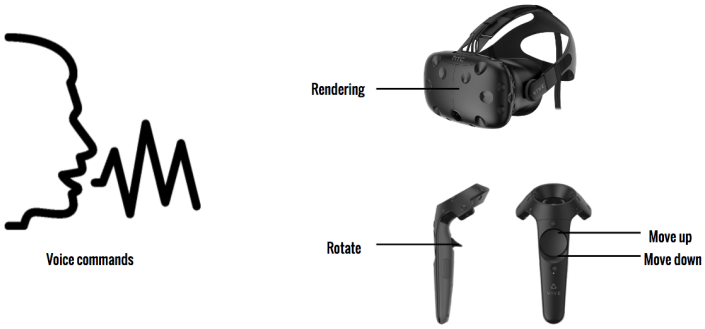
The tag suggestions are selected depending on the current shown pictures, trying to find similar tags present in most of them. At the beginning, the suggested tags are the most present in all the pictures tagged.

5.1.2 Advanced functionalities

The user can

- Query different tags via mathematical intersection and union
- Move the pictures closer and further (zoom)
- Rotate the selected picture
- Move up and down through the pictures

5.2 Interactions



Interactions with the applications are possible in four possible modes:

1. **Movement** Being in a Virtual Reality, the user can move around, if the space around him allows, and get physically closer to the pictures to analyze them better. [1].

2. **Vision** When a user looks at a picture, it gets automatically selected to allow the user to apply advanced functions. Whenever he looks at the ground, he will always find suggestions for tags.

3. **Speech** The user can say a tag, and immediately pictures associated with that tag will be shown. When the user says the keyword `help` a text of help is shown in front of him, so he can read the different commands he can say and execute.

   When he says the keyword `and` or `or`, followed by a tag, the related mathematical operation will be executed on the set of the currently shown pictures and the set of pictures associated to the newly pronounced tag. When he says `further` or `closer` the pictures are zoomed in or out according to the keyword said.

4. **Touch** Using the controller of the HTC-Vive the user can toggle different commands: using the back trigger he can rotate of 90 degrees the selected picture, in order to improve its visualization.

   Using the buttons up and down of the touchpad, he can move up and down in the cylinder of pictures that surrounds him, being able to see also the pictures too far from him.

Note on visualization The pictures are shown in the Virtual reality environment all around the user. This means that the user, in order to look at pictures, simply has to rotate. If many pictures are shown, they form a sort of cylinder around the user, being displayed in a circular way around him at different height level. That's why it was necessary to add commmands to move up and down.

During the development of the project we tried out different layouts of pictures: in our first prototype, in fact, pictures were shown in a big plane in front of the user. This means that with a big number of pictures, the user would need to move left, right, up and down, and could only look in the direction of the plan since doing only one layers of pictures wouldn't have allowed to see many pictures at the same time. We also tried different radius for the pictures circle and different curving shapes.

Movements left-right are allowed in virtual reality, since the user can move and the sensor will track his movement. But big sets of pictures would have meant that the user had to move probably way over its possible space around him, since the HTC Vive has physical limitations due to the cable attached to the computer and to the room itself. This is why we preferred to have the user surrounded by pictures, offering two buttons to move up and down that we would have needed to add anyway.

Note on the controller commands The three different commands controller-activated were also implementable by voice query, as most of the other com-

mands of the application. We preferred a controller-approach because it gives a better immediate feedback than the voice recognition, and because we didn't want to overcharge our dictionary with other keywords that the user had to remember.

5.3 Software Technologies

The selected technologies are :

- Google machine learning library TensorFlow for the neural network,
- SteamVR library to make Unity3D work with the HTC Vive,
- Unity3D for the HTC Vive Virtual Reality environment.
- C# on *mono* reduced set of Microsoft .net framework.
- Microsoft-Unity windows speech recognition library (`Unity.Windows.Speech`)for the voice commands.
- Json format to store tags associated to a picture path.
- `Newtonsoft.Json` library to parse the Json in the C# environment.

As said above, the VR will be built using the game engine *Unity3D* [1] for the application graphics and code. It has been chosen by considering different scientific papers such as [2] and [3].

The pictures tagging is done automatically using a neural network. More precisely the tagging is done using TensorFlow [4], a deep convolutional neural network, with a selected pre-trained model on ImageNet [2]. The selected model, Inception-v3, can classify entire pictures library into 1000 classes such as for example cliff, seashore, dishwasher, leopard, ... Each picture given as input to the network will give as an output a vector of probability corresponding to the probability to find each of the 1000 class in the pictures. Then the idea is to select the ones with the higher probabilities.

5.4 Architecture

The project is written using Unity3D, and is divided in two parts: the C# Unity display scripts and the C# application logic.

[1] https://unity3d.com/
[2] http://www.image-net.org/

5.4.1 Unity Scripts

Files are located in src/ImageDisplayer/ImageDisplayer/Assets.

The Unity scripts handle the visualization and the interaction with the user. There are three scripts:

- `Main.cs` handles the scene, all the game objects, the disposition of the pictures, the texturing and all the commands. It has a specific method for each command, and it handles the current pictures, deleting them from memory when they are not shown anymore in order to prevent memory leaks.
- `DictationScript.cs` initializes a grammar with the tags and the keywords. With that it loads a `KeywordRecognizer` that listens to the told keywords. When recognized, it will call the right method on the `Main` script. If the recognized word is one of the commands-keyword, it will call the relative functionality in the main. Otherwise, it will call the main for loading the pictures.
- `WandController.cs` handles the controller input. It calls the rotate on the main script if needed, otherwise it moves the main camera in case of movement.

5.4.2 Application Logic: Manager

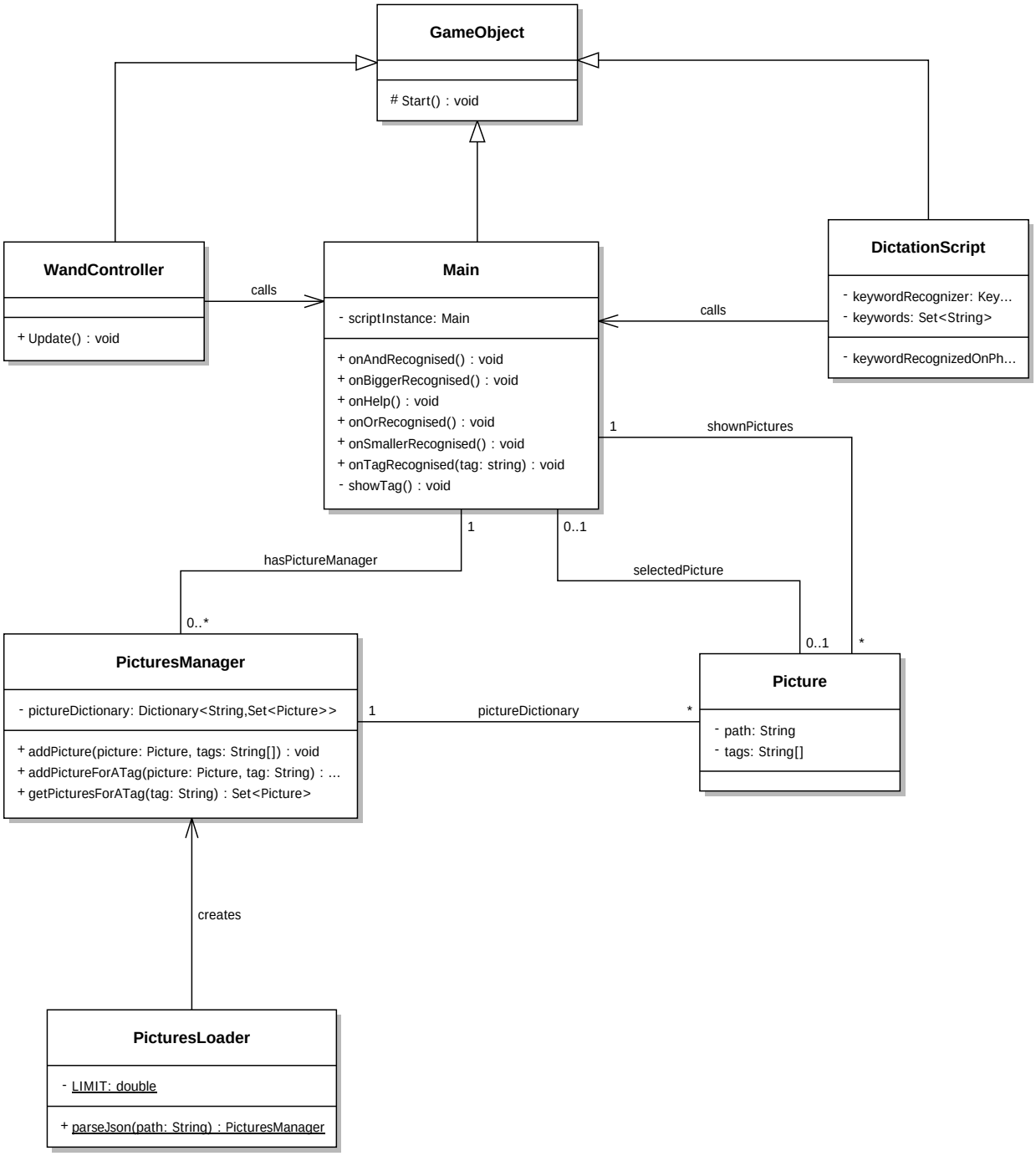
Files are located in src/ImageDisplayer/ImageDisplayer/Assets/Manager.

The files on the Manager namespace handle the representation of pictures, hold the reference to them and provides the querying.

It is composed by three classes, stored in three different files:

- `Picture.cs` represents a `Picture`, with its filepath and tags.
- `PicturesLoader.cs` provides functionalities to parse the JSON containing the pictures paths and tags. A tag is accepted only if the confidence level (calculated by the neural network) is higher than 35%. The rest of the tags are ignored. It uses the `Newtonsoft.Json` library to ease the parsing of the JSON.
- `PicturesManager.cs` manages the set of pictures, allowing to add and query the set. The set of pictures is stored as dictionary, with the tag string as key and the `Picture` object as value.

UML Diagram In the next page you can find an UML diagram of our architecture. On the top part you can see the Unity scripts, on the bottom part the Manager.


GameObject
Start() : void
WandController
+ Update() : void
calls
Main
- scriptInstance: Main
+ onAndRecognised() : void
+ onBiggerRecognised() : void
+ onHelp() : void
+ onOrRecognised() : void
+ onSmallerRecognised() : void
+ onTagRecognised(tag: string) : void
- showTag() : void
calls
DictationScript
- keywordRecognizer: Key...
- keywords: Set<String>
- keywordRecognizedOnPh...
1
shownPictures
1
hasPictureManager
0..1
selectedPicture
0..*
PicturesManager
- pictureDictionary: Dictionary<String,Set<Picture>>
+ addPicture(picture: Picture, tags: String[]) : void
+ addPictureForATag(picture: Picture, tag: String) : ...
+ getPicturesForATag(tag: String) : Set<Picture>
1
pictureDictionary
*
0..1
*
Picture
- path: String
- tags: String[]
creates
PicturesLoader
- LIMIT: double
+ parseJson(path: String) : PicturesManager

Note: External Source Code In the source code are also present classes of `SteamVR`, that are necessary to deploy on the HTC Vive. Those classes are not part of this project.

5.5 Challenges

We faced many challenges, mainly because nobody in the group had previous experience with Unity nor C#. Unity uses a reduced set of the `Microsoft .net Framework 3.0`, so many libraries where incompatible, or some functionalities of C# couldn't be used. Furthermore, Unity is not compatible between different versions, so we all needed to have the exact same release, and this got us into many troubles and time loss before we found out what was the problem.

There is poor software interoperability between Unity and its library, and we were forced to use the Microsoft Speech Recognition Library, that works only on Windows. Since we are all Linux or mac users the initial goal for us was to make the application run on any of those machine, but Windows was needed. For those of us that didn't have Windows 10 an update was required, since the libraries work only with that version of windows.

The only working speech recognition library we found works only on keywords. This means that you cannot speak naturally to it, and tags need to be initialized to a specific grammar. We used all the possible keywords given by `Tensorflow`, but when no word is recognized there is no feedback for the user to know if he just misspelled or if that tag is not present in the system. We also had problems with our accents, since we all are not native English speakers and we all have our different accent.

Working with Unity3D along with SteamVR also made us face a lack of documentation, since it's new technology and many library or components don't have a really complete documentation, and many things had to be found by experimenting.

Finally, in order to run our application using the HTC Vive headset, we needed to use a really modern and powerful desktop computer, because our laptops couldn't render the VR environment smoothly.

6 Evaluation

In addition to continuous feedback from the teaching team and others students, we did a formal evaluation.

This evaluation aim is double; it first assess the usability as well as the user experience.

Dealing with limited means, our evaluation is limited to five volunteer evaluators with no previous knowledge of our product. It was a challenge to find more evaluators due to the fixed location of the VIVE setup.

6.1 Usability

Usability evaluation can be defined as trying to assess the ease of use and learnability of our interface and moreover trying to assess that the usability goals are met. Due to the limited number of evaluators, the Nielsen Heuristic seemed appropriate.

6.1.1 Conducting evaluation

The evaluation is a discount evaluation of 5 users using Nielsen Heuristic. According to Nielsen and Landauer (1993), this evaluation should allow us to discover about 75 percents of usability problems. Also according to Nielsen and Landauer the cost to benefit ratio of having 5 users is nearly maximal, being just under 60.

This was a two passes evaluation during which we observed and took notes about an evaluator. Each evaluator passed the evaluation one by one without seeing each other to avoid getting information about how to use the interface. The idea here was to let the user use the application without explanation and see if the user managed to learn how to use it.

We then asked the evaluators to fill a form regarding each heuristic and then did a debriefing with them one by one.

6.1.2 Results

The observation concluded that all the 5 users were able to turn their head around and explore pictures after saying a tags without looking at the help feature. Some had to read the help to use other features but some got the moving part and recommendation directly. Then all of them read the help and used most of the features. On the "could be better" side, some people with strong accent were poorly recognized by the voice recognizer. One user was reading what he was seeing on the screen out loud making the system recognize unwanted tags.

Now regarding the heuristics, the 5 evaluations could be summarized as follow :

- **Visibility of system status** : Since the pictures of a particular tag are displayed, it's easy to know the system status. Could immediately tell when a tag is recognized or when it failed to recognize.
- **Match between system and the real world** : Saying keyword is a good match between system and real world. One of the evaluation subject said that the use of the word "tag" is too system-oriented.
- **User control and freedom** : Not really applicable here since the last tag enable the user to go back to that particular tag.

- **Consistency and standards** : Every action has its own clear purpose.
- **Error prevention** : No real error can be made, same as user control and freedom.
- **Recognition rather than recall** : Help needs to be read to find advanced features.
- **Flexibility and efficiency of use** : All evaluators left this one blank.
- **Aesthetic and minimalist design** :Beautiful and minimalist design. Nothing is not in its right place.
- **Help users recognize, diagnose, and recover from errors** : Nothing happens in case of an error (just wrong pictures shown). So not really applicable. More random suggestion could be needed.
- **Help and documentation** : Help is clear even dense.

6.2 User experience

User experience refers to the evaluator emotions and attitudes about using our interface. The idea behind this evaluation is to assess how the user feels about our interface and assess that the user experience goals are met.

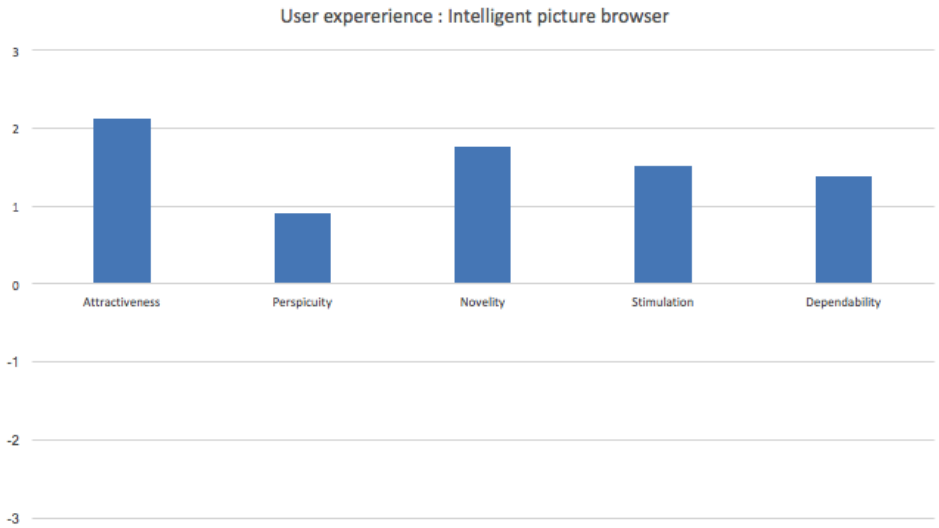
6.2.1 Conducting evaluation

This user experience evaluation strictly follows the one given by the assistant during the lab session for the spoon. We used some help from [5] to conduct it properly. This evaluation give us a score between -3 and 3 for the following keypoints:

- Attractiveness
- Perspicuity
- Novelity
- Stimulation
- Dependability

It was done just after the usability evaluation. There are 5 evaluators in total so we have to keep in mind that the result could be not statically significant.

6.2.2 Results



The result shown on the bar graph above tends to best part of the interface is its attractiveness and a point that could be improved is its perspicuity. The insight behind this graph is could be described as follow :

- It's globally enjoyable/pleasing
- It's exciting
- It's innovative
- Could be easier to learn
- Could be more predictable
- Could be faster

7 Acknowledgement

Before ending this report, we would really like to thank the VUB Soft Lab for lending us the HTC Vive needed for the realization of this project. We also would like to thank participants of the evaluation and everyone who gave us feedback during the making of this Next-gen UI.

8 Conclusion and Future work

We proposed IPB, an intelligent picture browser that allows the user to see pictures corresponding to tags all around him in a VR environment. This

was accomplished in several iterations using state-of-the-art technologies. The outcome is fully functional and was tested against different users with different accents to assess their user experience as well as the application usability. The evaluation was mostly positive but flaws were discovered thanks to the feedback we received and most of them were corrected. However, the application could be, in the future, made easier to learn as well as faster to run.

References

[1] A. V. S. Calado, M. M. Soares, F. Campos, and W. Correia, "Virtual reality applied to the study of the interaction between the user and the built space: A literature review," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8014 LNCS, no. PART 3, pp. 345–351, 2013.

[2] T. Mazuryk and M. Gervautz, "Virtual Reality: History, Applications, Technology and Future," *Technical Report*, vol. TR-186-2-9, 1996.

[3] A. V. S. Calado, M. M. Soares, F. Campos, and W. Correia, "Virtual reality applied to the study of the interaction between the user and the built space: A literature review," in *International Conference of Design, User Experience, and Usability*, pp. 345–351, Springer, 2013.

[4] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[5] B. Laugwitz, T. Held, and M. Schrepp, "Construction and evaluation of a user experience questionnaire," in *Symposium of the Austrian HCI and Usability Engineering Group*, pp. 63–76, Springer, 2008.