

ÉCOLE POLYTECHNIQUE  
FÉDÉRALE DE LAUSANNE

# Human Perception Of Guided Interaction

*Ecole Polytechnique de Lausanne*

*June 2015*

Semester Project  
Jad Khoury

Supervisors:  
Ronan Boulic & Henrique Galvan Debarba

1 - Introduction	2
2 - Related Work	4
3 - Implementation	5
3.0 - Keywords	5
3.1 - Virtual Environment	6
3.2 - Distortion Model	8
3.3 - Code Implementation	13
3.4 - Experiment design	15
3.5 - Measurements	17
3.6 - Hardware precision	19
4 - Results and Discussions	21
5 - Conclusion	25
6 - References	26
7 - Appendix	27

# 1 - Introduction

When producing a movement, two distinct parts of the brain work separately. It enables subjects to distinguish sensorial stimuli resulting from self-produced movements to the ones created by an external part; along with the correctness of a self-produced movement. To achieve this, one part of the brain dictates the movement to the motor system, then recover the real sensory feedback. The other part makes a copy of the intended movement, called efference copy, and computes an estimated sensory feedback based on this efference copy. The brain then compares these two feedback to make the above-mentioned judgement (fig 1.1). Experimental studies have shown that the primary sense used to make this comparison is the vision.

Laying on this concept, we can wonder how a subject reacts when his actual sensorial feedback is manipulated when producing a self-produced action. To answer this question, experimental researches were designed to study the various effects of modifying one aspect of the visual feedback of a subject when producing a movement: precision and speed ("throughput") of the movement, and awareness of the distortion. As a result, it occurred that for some degree of distortion, the subject actually thinks that the modified movement suggested by the manipulated visual feedback is the one he performed.

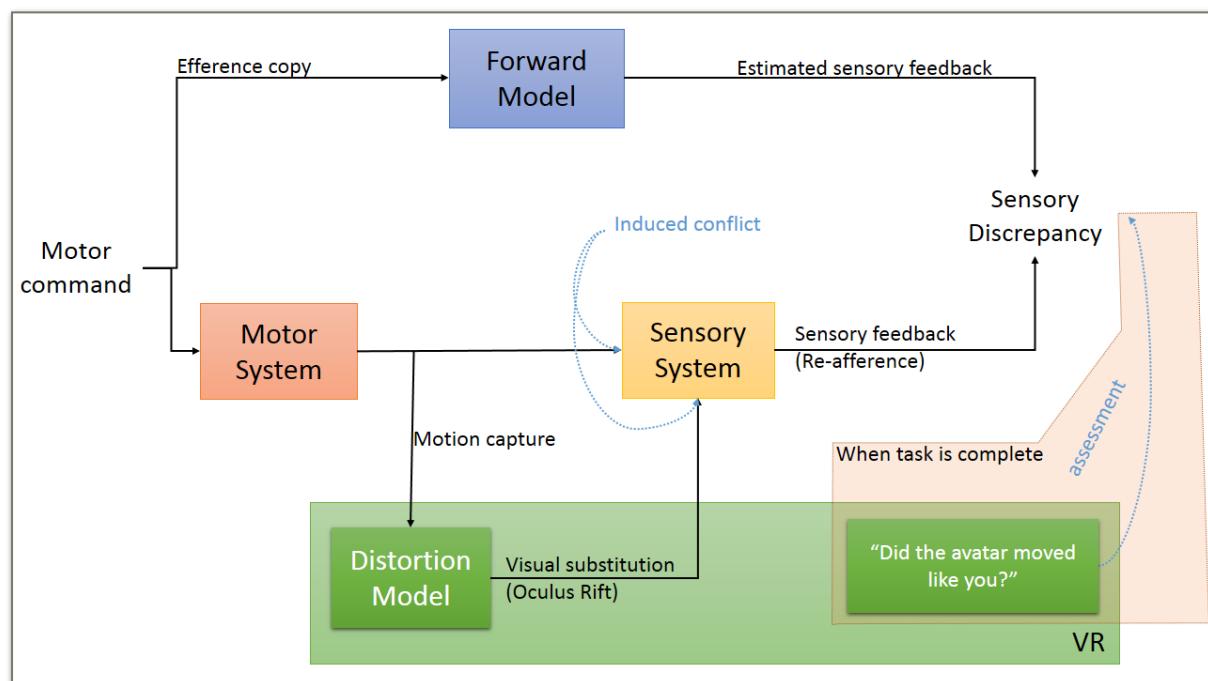


fig 1.1 Redirect motion concept

Our project intends to analyse the consequences of applying a distortion to the seen movements when doing a multi-directional reaching task, defined on ISO standard. More specifically, we study the effects of applying a positive distortion, that helps the subject completing the task, and of a negative distortion, that makes the task harder. During this experiment, the subject sits on a chair wearing an HMD, facing an object that will be used as a display (fig. 1.2). In the rendered scene, he will see displayed on the object a cercle of targets, that will successively be highlighted in a specific order (fig. 1.3). He will then have to touch the highlighted target while a distortion is applied or not to the visual feedback of his movement.



fig. 1.2 - Subject going through the experiment

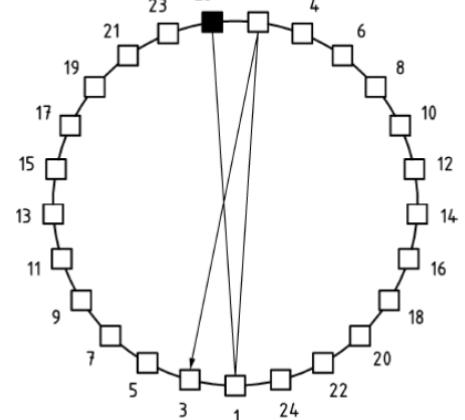


fig. 1.3 - Reaching order

## 2 - Related Work

Redirected motion is a topic already studied under many forms: redirected walking, pointing etc ... The redirection of a multidirectional reaching task in particular has been studied by Luv Kohli, Mary C. Whitton and Frederick P. Brooks Jr. from the University of North Carolina. In their first project, they study the redirection of a hand movement that compensates an angular difference between a real-space object and its virtual-space rendering (fig 2.1). The goal of their study was to evaluate the effectiveness of warping real world object in a slightly different virtual space in order to provide a haptic feedback corresponding to the rendered scene without needing great precision on the rendered object position. It is focused on the throughput of the subject when performing the task, i.e. the number of reaching by unit of time. They concluded that this redirection didn't have a great impact on the subject performance (speed) and on their error rate (accuracy). A second experiment was designed to investigate the ability of a subject to train and adapt to the distortion that compensate the angular displacement between the real world object and the rendered object.

Our work differs from these researches in the fact that we focus our study on the perception thresholds, aspect that has only been superficially approached until now. Furthermore, our distortion is of a different nature since it is intended to make a reaching task easier or harder.

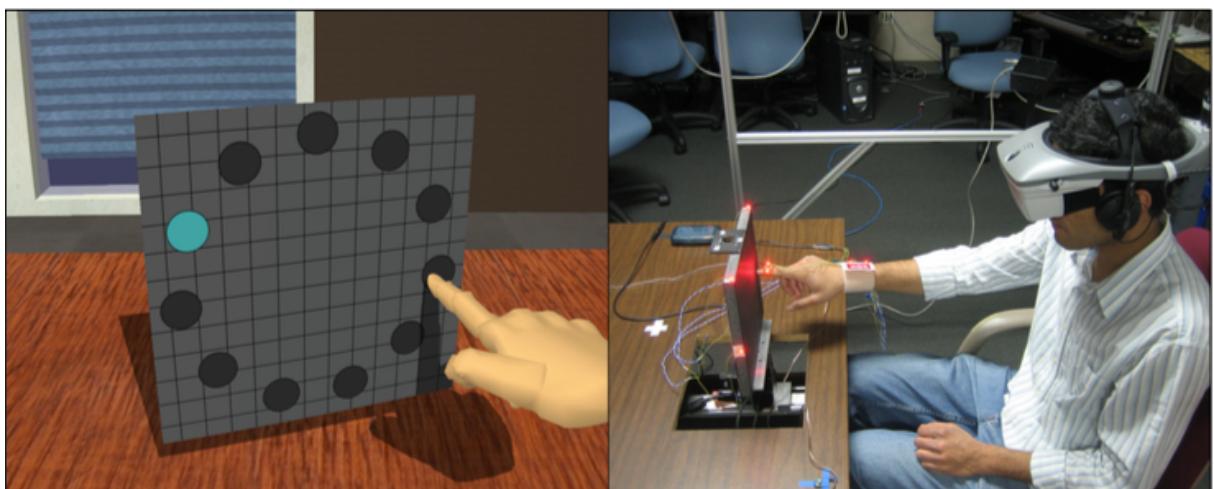


fig 2.1 : A user touches a virtual board that is oriented differently than the real board providing passive haptic feedback

# 3 - Implementation

## 3.0 - Used Keywords

- ~ Target: rendered coloured disc the subject has to reach during the multi-directional task.
- ~ Virtual distance to target. Distance from the displayed redirected hand to the displayed target.
- ~ Distance to target. Distance from the real hand to the rendered target.
- ~ Action Range: range around the current target in which the distortion is effective.
- ~ Target radius: radius of the displayed target disc.
- ~ Virtual target radius: radius around the target in which the real world hand is redirected above the virtual target. In other words, if the virtual target radius is 5 and the target radius is 2, when the real hand is at 4.5 from the center of the target then the rendered redirected hand is already above the target.
- ~ Reciprocal of the function f: function  $g = f^{-1}$  such that  $f(g(x)) = g(f(x)) = x$
- ~ Unity, Unity Game Engine: “Unity is a cross-platform game engine developed by Unity Technologies and used to develop video games for PC, consoles, mobile devices and websites.”<sup>1</sup>
- ~ One reaching: the act to move one’s hand from its position to the activated target.
- ~ One trial: Set of successive reaching in a particular order necessary to activate all targets.
- ~ One block: Set of trials, with predefined length and randomly distributed chosen parameters.
- ~ id: factor used to compute the radiiuses
- ~ dId: distortion id, factor used to compute the virtual radius

---

<sup>1</sup> source: Wikipedia

## 3.1 - Virtual Environment

The rendering of the virtual environment is computed with the Unity Game Engine. This choice allows dynamic scene creation and easy script implementation. The scene designed for this experiment is extremely simple since the tasks only require to render a screen on which some targets will be displayed. For this project, we render a neutral virtual environment to allow maximum focus and minimum distraction for the subject (fig. 3.4). The body is not represented, neither the head. Only the hand is rendered with a minimalist 3D Object (fig. 3.2) and the display on which the targets and the informations are displayed is a simple grey box (fig. 3.1).

The targets are stucked to the display and represented by coloured discs. When a target activates itself, it becomes orange, and then only this target can be triggered.

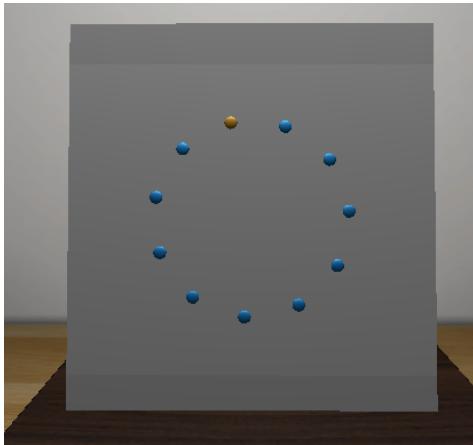


fig 3.1 : the display and the targets

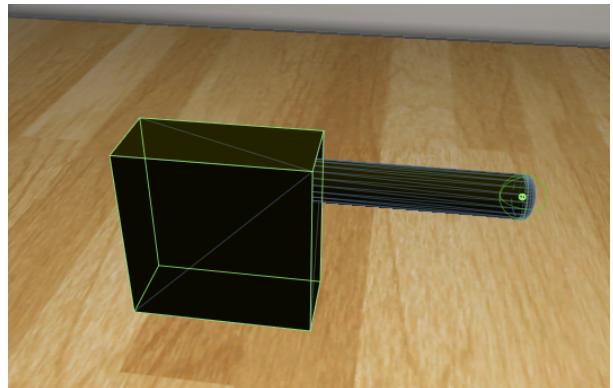


fig 3.2: the rendered “hand”

All the objects of the real world scene are being tracked using motion capture markers. This ensure the rendered objects are always at the appropriate position and that if one of the object is moved, for example if the subject moves the display, the rendered object will still represent the real position of the object. All objects are pre-calibrated at the exception of the display which is calibrated for each subject, to ensure the length of the index, the way the glove it put etc ... do not interfere with the reaching precision. On the hardware point of view, this tracking is performed by the PhaseSpace motion capture system (fig 3.3). The LED Markers of the tracked objects (HMD, hand, display, table) are captured by the PhaseSpace cameras (1), which transfer the captured position to the server (2). The server then

compute the 3-dimensional position of each marker from distinct 2D positions he received from the cameras having the marker in their field of view. Afterwards, the server sends the 3D position of each marker to the VRPN client on the computer via the VRPN link established (3). The MocapInputController script transmit these datas from the client to the Unity Game Engine (4) which sets the rendered object position accordingly to the marker position and the calibration of each object. Finally, the rendered scene is displayed on the Oculus HMD via a simple HDMI connection (5).

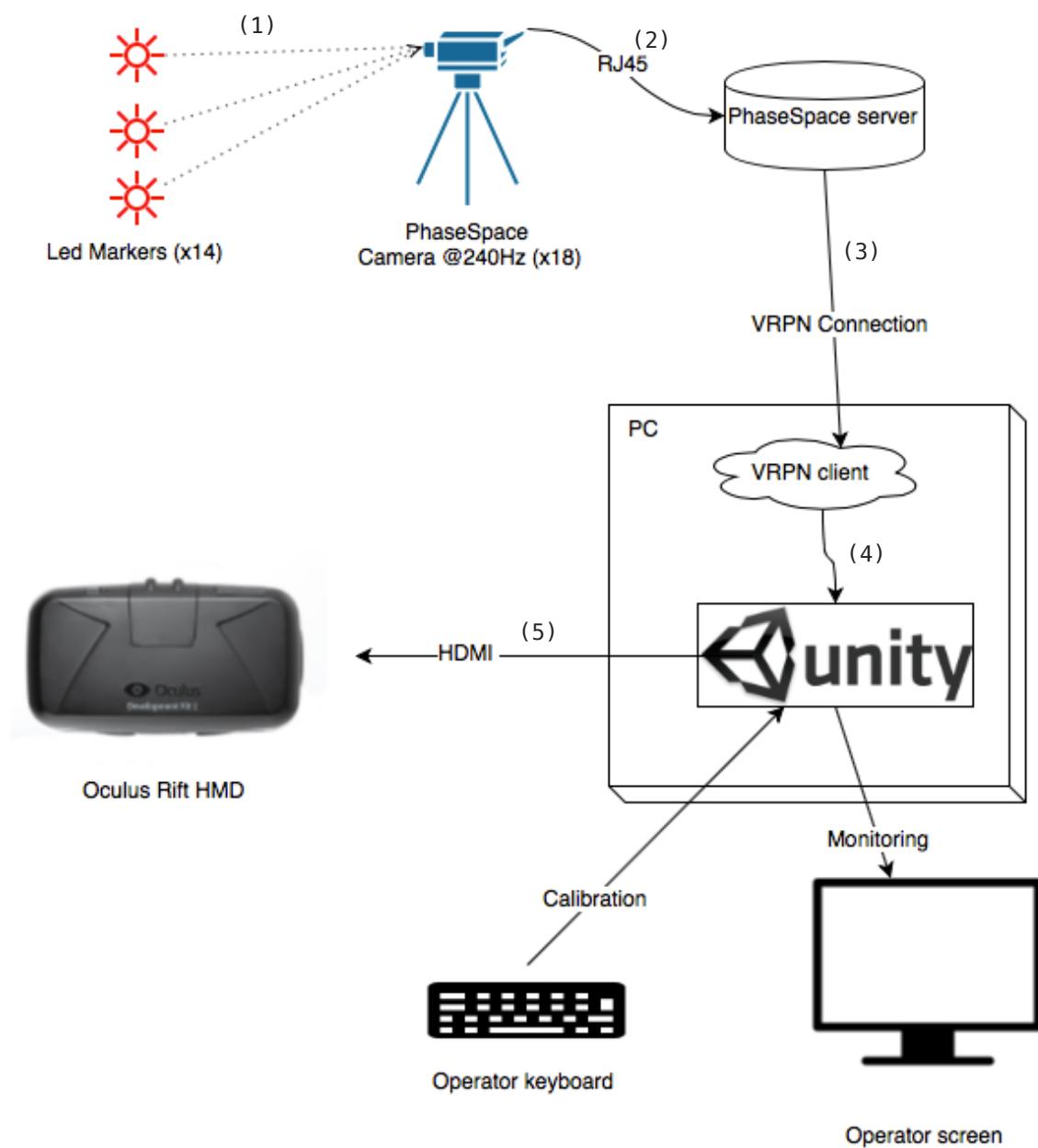


fig. 3.3 - Hardware setup

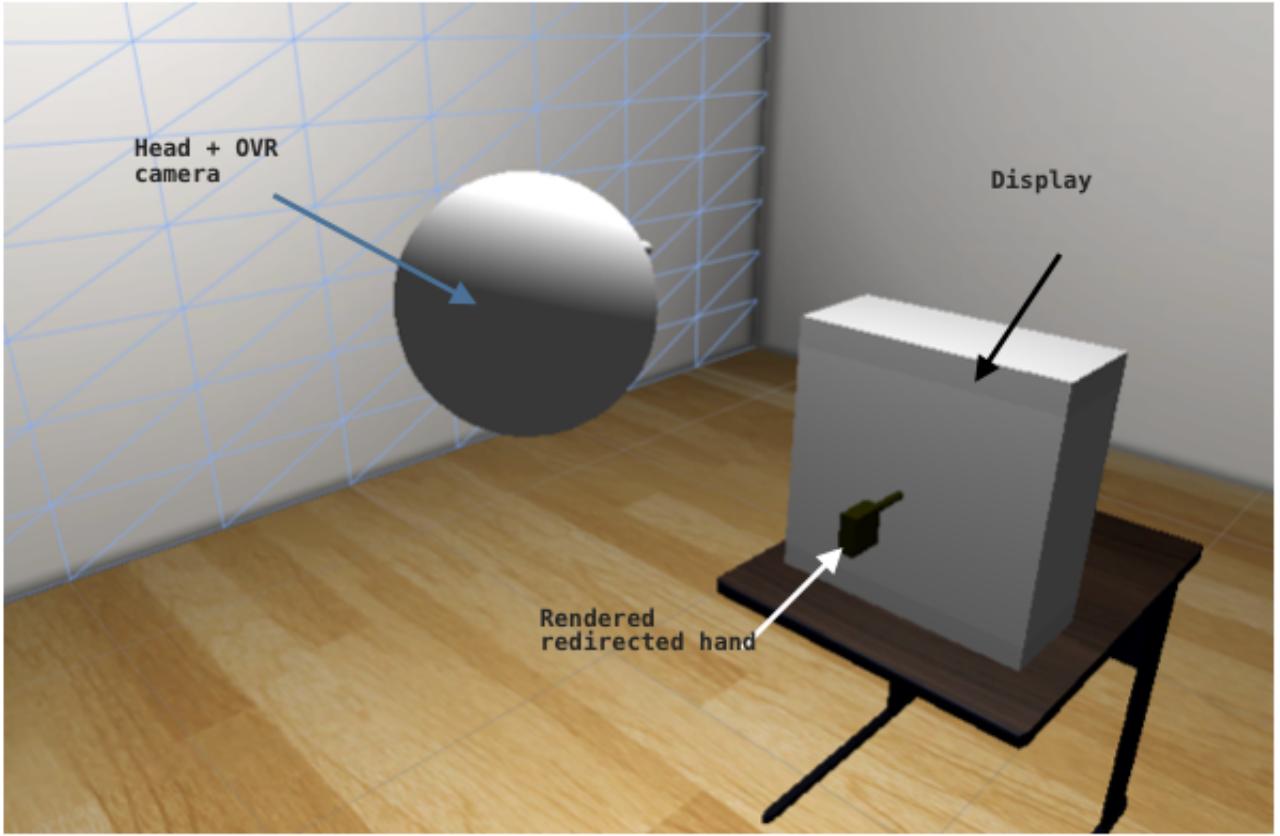


fig. 3.4 - Rendered environment

## 3.2 - Distortion Model

During the following discussions, the virtual distance to the target will always be equals to the real distance to the target when the hand is outside action range. I.e.

$$vD = \begin{cases} d & \text{if } d > aR \\ \text{distort}(d) & \text{if } d \leq aR \end{cases}$$

The distortion model is the function (in the mathematical meaning) computing the distance from virtual hand to the target in function of the real distance between hand and target. Since the task is a purely reaching task, we chose to compute everything in term of the distance to the target (dTT). Therefore, the visual feedback of the hand will always be displaced towards or against the target, and in no other direction. Moreover, we chose to isolate the distortion on the target plane, i.e. if the subject moves his hand forward and against the target, orthogonally from the plane on which they are displayed, no distortion will be applied, but if his hand stays at 1cm over the display and he moves his hand parallel to it, then the distortion will be

effective. The predefined distortion model used a gaussian derivative that suffered numerous problems: hard to give physical meaning to the parameters and difficulty to control the steepness of the curve. Indeed, for the distortion to be valid, the values of vD should be unique, and so the function should be strictly monotonically increasing or decreasing. To ensure this, the parameters need to be constrained in such way that a particularly strong steepness does not result in a non-monotonous distortion function.

The first distortion model designed to take its place used a simple gaussian function with parameter defined to be easily interpreted and protecting the redirection from the above mentioned steepness problem. It was defined as follow:

$$vD = d - \frac{\text{strength}}{100} \cdot aR \cdot \left( 4 \ln\left(\frac{p}{a}\right) \cdot \left( \frac{\left(x - \frac{aR}{2}\right)^2}{aR^2} \right) \right)$$

- . a = amplitude (strength) of the curve
- . p = precision, eq. standard deviation
- . aR = action range
- . d = real distance to target
- . vD = virtual distance to target
- . For negative distortion, vD = d + ...

There were still a major problem that shows on the plot of the function (fig 3.1): the behaviour was good when entering action range but when getting really close to the center of the target we lose the increased accuracy. Unfortunately this issue make this function useless since it cancels the main goal of the distortion function: making the movement more/less precise around the center of the target.

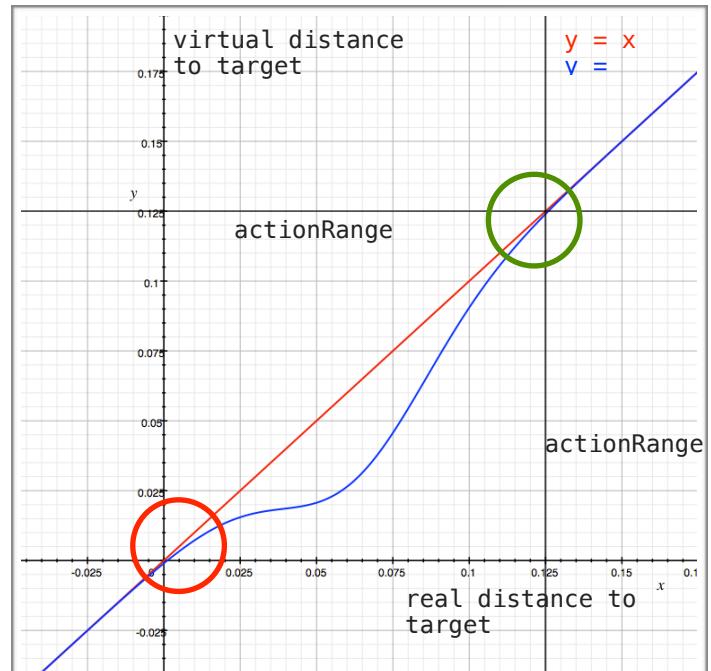


fig 3.1 - Gaussian implementation of the distortion function. Here,  $p = 0.001$ ,  $aR = 0.125$  (m), strength = 100

No matter how much the parameters were tweaked, the needed behaviour around zero could not be obtained.

So another distortion model was created in term of trigonometric functions:

$$vD = d - \frac{\text{strength}}{100} \cdot \frac{aR}{\pi} \cdot \sin\left(\frac{\pi \cdot d}{aR}\right)$$

- .  $d$  = real distance to target
- .  $vD$  = virtual distance to target
- . For negative distortion,  $vD = d + ...$

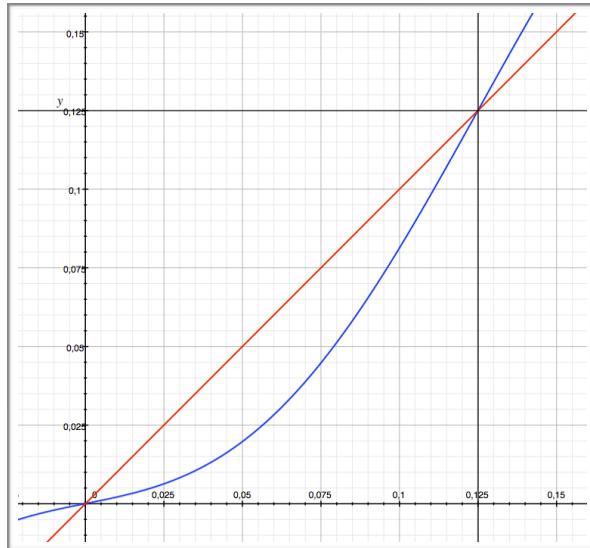


fig 3.2 - Sinus implementation of the distortion function. Here,  $aR = 0.125$  (m), strength = 80

The main advantages of this function was that there were only two parameters (action range and strength), that the behaviour was really good in positive distortion and that is was simple to implement.

However, a conceptual problem with the first two function is that they used an arbitrary **strength** variable to define the actual strength of the distortion applied to the position. In order to be more meaningful, the function should be defined not in term of an arbitrary variable but in function of a displayed and virtual target radius, themselves computed using standard formulas explained later on. To be correct in regard to the virtual and real target radius, the distortion function should respect the constraint  $\text{distort}(vTR) = tR$ , i.e. when the real hand is in the range of the virtual target radius, the displayed hand is already above the displayed target radius. Based on this constraint, the strength became computed as

$$\text{strength} = \begin{cases} \frac{\pi \cdot (vTR - tR)}{aR \cdot \sin\left(\frac{\pi \cdot vTR}{aR}\right)} & \text{if } \text{result} < 1 \\ 1 & \text{else} \end{cases}$$

This allowed to keep the function as it is, but compute the strength as a function of the radiiuses.

Unfortunately, when computing  $d +$  distortion, it didn't return the reciprocal function as needed. Fig. 3.2 shows that the function is not asymptotical to  $y$  when approaching 0, when it should be since the positive distortion is asymptotical to  $x$  when approaching 0. Furthermore, because of its construction, it was impossible to find the mathematical reciprocal function.

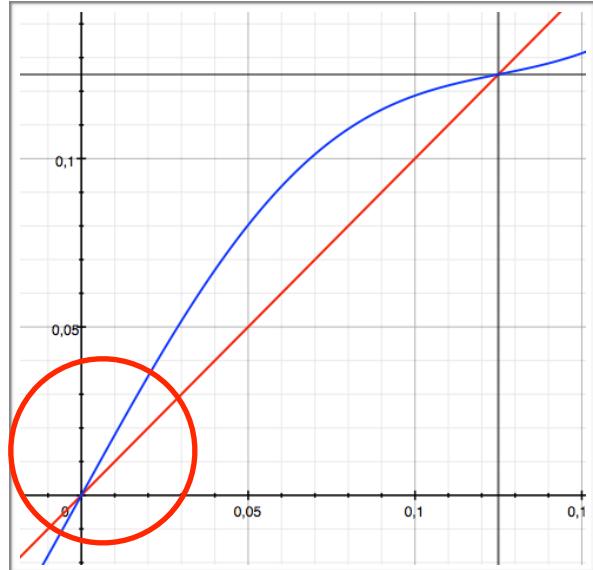


fig 3.3 - Sinus implementation of the distortion function.  
Here,  $vTR = 0.015$  and  $tR = 0.05$

The needed distortion function was then a function defined in term of  $tR$  and  $vTR$ , that is its own reciprocal function when set as negative distortion:

$$vD = aR \cdot \left( \frac{d}{aR} \right)^{\log_{\frac{vTR}{aR}} \frac{tR}{aR}}$$

- $vD$  = virtual distance
- $d$  = real distance
- $vTR$  = virtual target radius
- $tR$  = target radius

We see that the redirected distance is not computed in term of  $d - \{...\}$  anymore. This has for consequence to allow passage from positive to negative distortion solely by having a  $vTR$  smaller or bigger than  $tR$ . Furthermore, the negative distortion behave as needed as seen of fig. 3.4

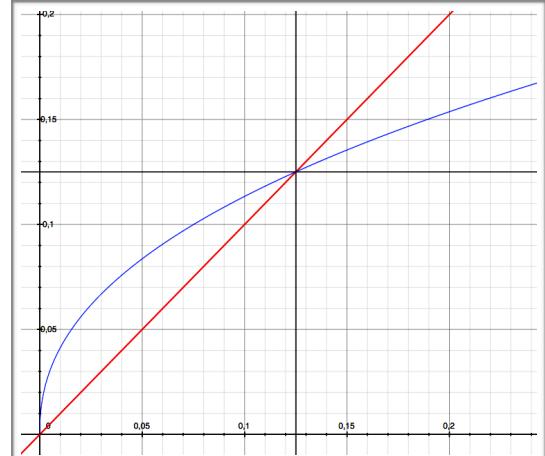
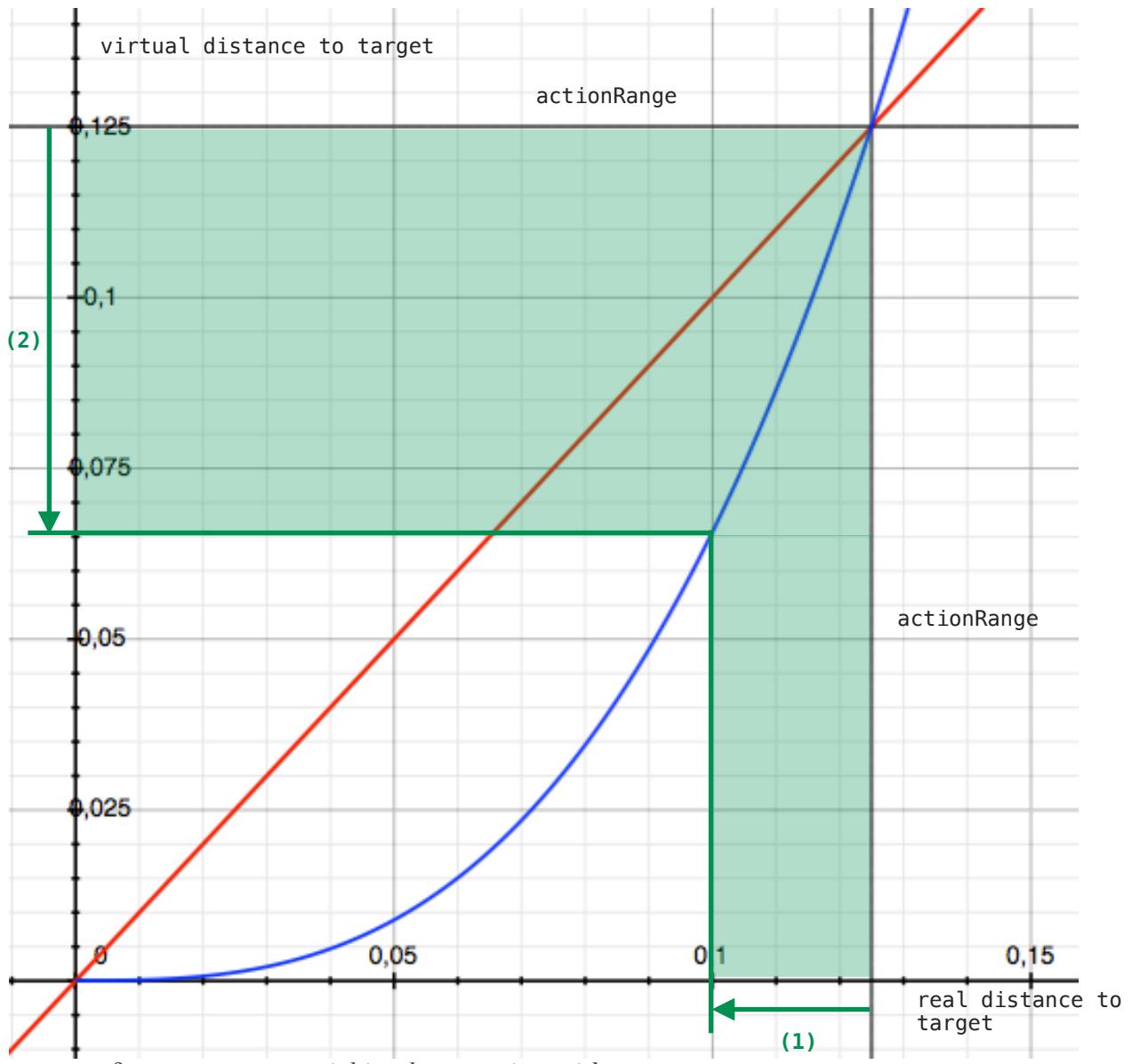


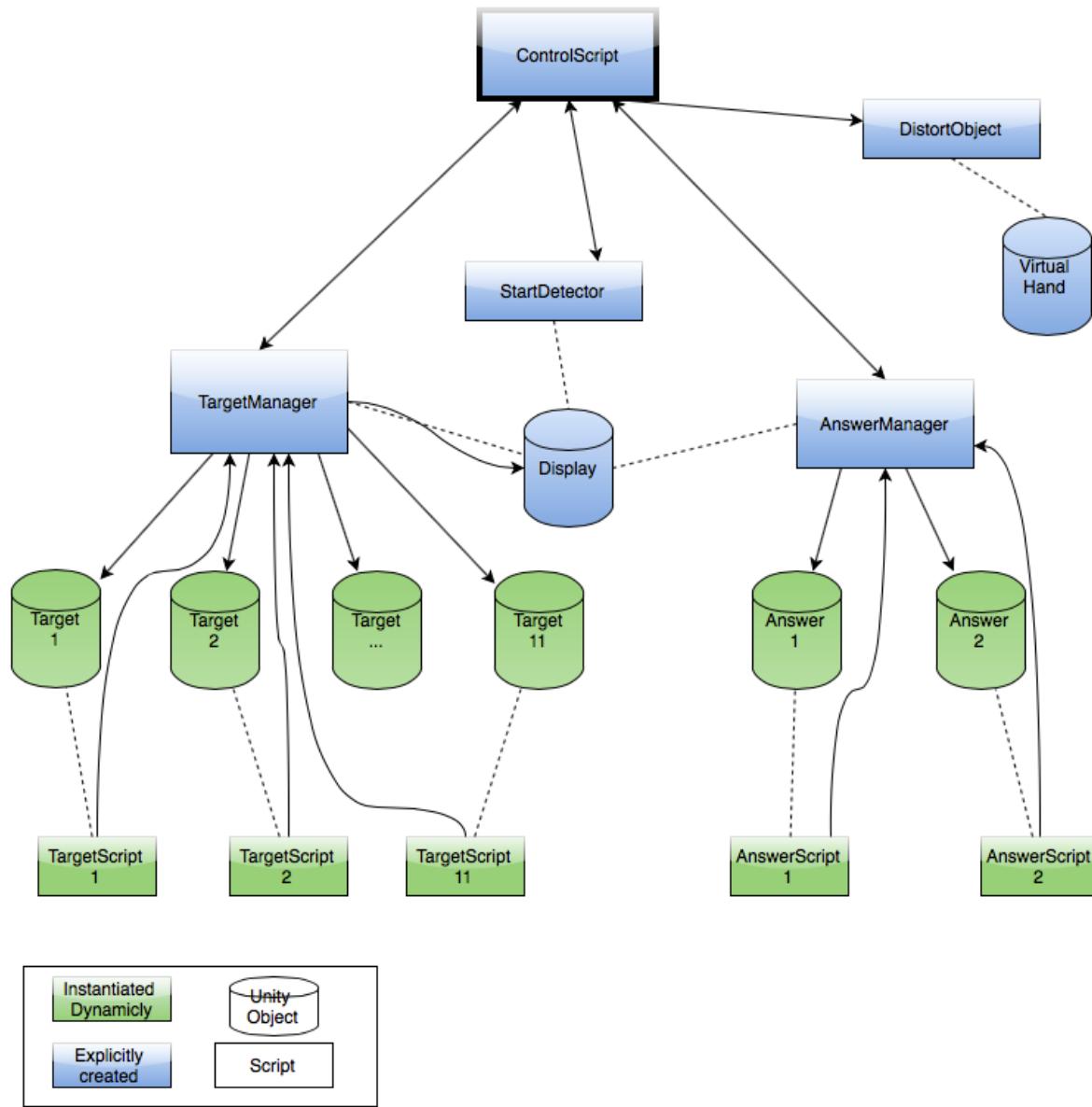
fig 3.4 - exponential implementation  
with  $vTR = 0.01$  &  $tR = 0.04$

To better understand these plots (fig. 3.5), we have to imagine that by going from right to left on the x-axis, we are approaching our real hand to the target. And the values given by the y-axis are the distance covered by the virtual hand. We see that in the first part of the movement, when the real hand covers a small distance (1) the rendered hand object covers a greater distance (2), resulting in an acceleration. But when approaching the target, the effect becomes the opposite (curve steepness going under  $45^\circ$ ): when the hand covers a great distance, the virtual hand only moves a little, resulting on a slower motion. The goal of this distortion is to offer a greater precision around the target, in addition to “attracting” the hand to it. The concept is exactly the same for negative distortion, but instead of increasing the precision around the target we decrease it (fig 3.4)



### 3.3 - Code Implementation

Unity can compile and run scripts in C# and Javascript. This project has been coded in C#. For this project, it was necessary to have a script controlling the whole experiment, in order to centralise the parameters and to ensure there is no code duplication in different scripts.

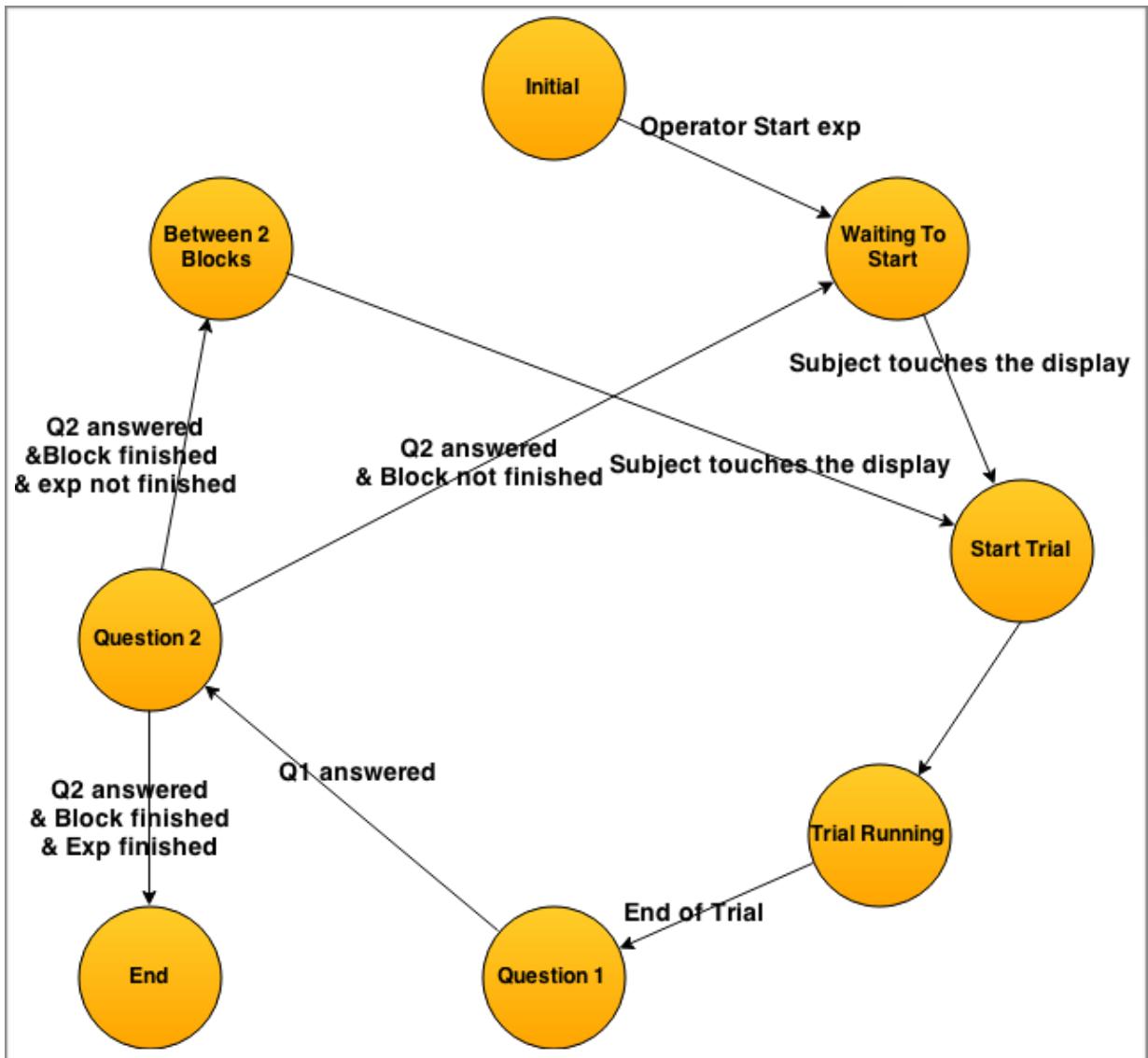


This project has been coded using C# scripts instead of in-editor Unity features (like animator etc ...), in order to be expandable and long-lasting. Indeed, Unity features evolves quickly, some are added, some become deprecated, and the designs created for this experiment should not become unusable after a short period of time. Furthermore, a lot objects and scripts

are created and attached dynamically to widen the possibilities the scripts offer and to be re-usable in different applications.

- The TargetScript is a script attached to each target, that is used to activate/deactivate the target, to listen to collision events, and on some condition pass the event up to the TargetManager.
- The TargetManager script is attached to the display and is used to create dynamically the targets GameObject, to set them up as needed (colliders etc ...) and to attach the required scripts. In addition to that, it listens to target events and when confirmed, switch the activated target to the next one. It also ensure that the distorted hand does not jump from the distorted position to the undistorted one by switching the distortion direction at a strategic time.
- AnswerManager and AnswerScript respectively behave more or less like the two above mentioned files at the difference that they do not have to manage redirection.
- StartDetector is a simple script that listen to collision events on the display and pass them up to the ControlScript if the latter is waiting for an event to start a trial
- ControlScript is the “brain” of the experiment. It stores all parameters necessary for the experiment, controls the other scripts and more importantly, implements the state machine used to ensure the experiment is happening in the right order.

When adding more and more feature and steps to the experiment, the control script became extremely unclear and prone to failures due to the triggers system used to manage the experiment. To remediate to this problem, the whole experiment flow has been implemented on a well defined state-machine. More often used in low level languages, the state machine concept offers an additional reliability in the sense that it does not allow the program to be in any other state than the predefined ones. And switches from one state to another are only allowed when a series of conditions are met.



## 3.4 Experiment Design

The final experiment has been designed such that it behaves like a stand-alone programme, or a video-game. From the moment the operator starts the experiment, the subject is completely autonomous. He decides when to start the trial, at the end he answers to the questions, he can take a break or not before starting the next trial etc ... All of the experiment is controlled by button-like interaction in the rendered scene. As shown on fig. 3.6 - 3.7, to answer a question the subject simply has to touch one of the two answers. The beginning of a trial also relies on collision event with the screen (fig 3.8).

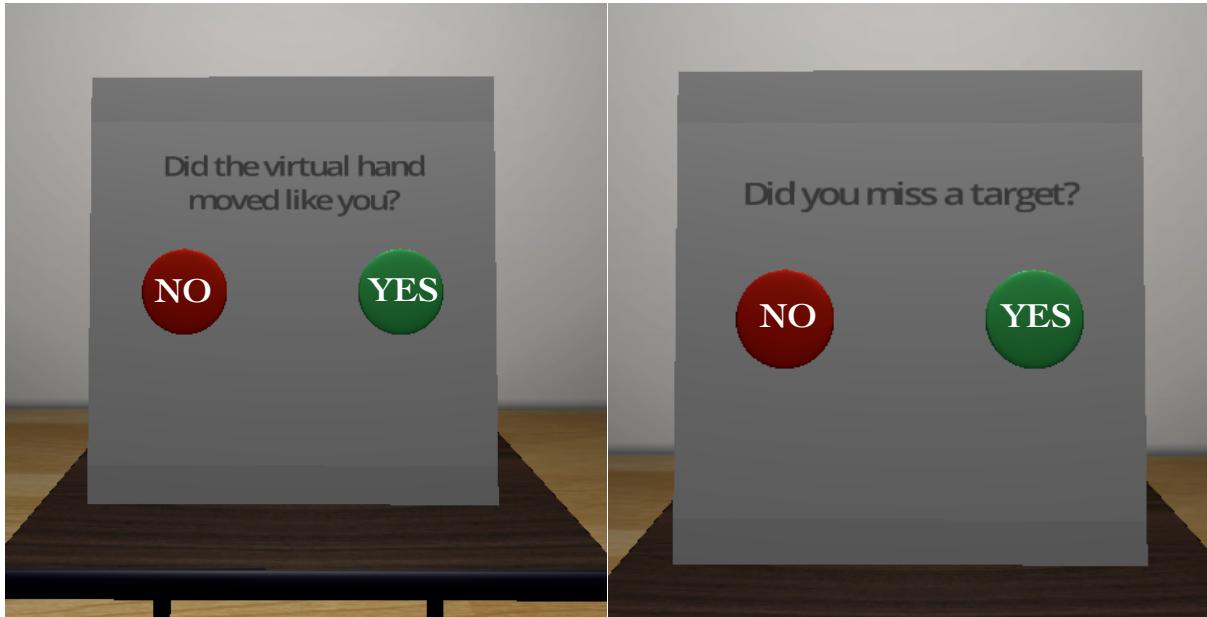


fig. 3.6 - Question 1

fig. 3.7 - Question 2

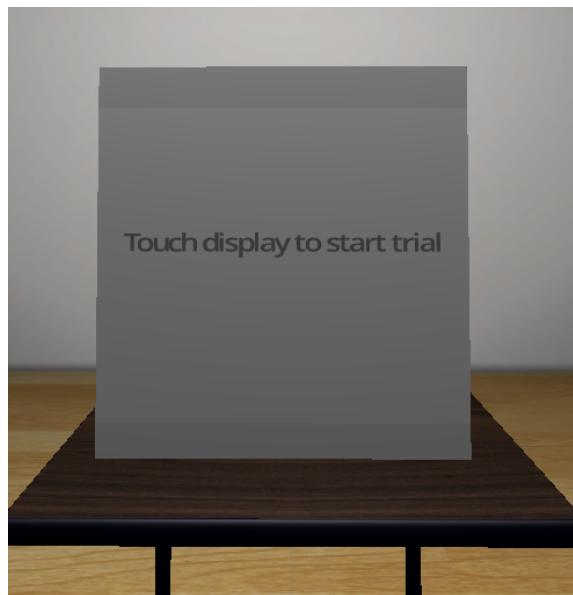


fig. 3.8 - Start display

This experiment relies on a set of pre-defined parameters:

- The number of blocks
- The number of repetitions of a parameter couple per block (explained later on)
- The action range
- The number of targets: gives the possibility to expand the experiment. We chose 11 since an odd number results in an equal distance between two opposite targets
- The distance between two opposite (therefore successive) targets: used to compute the circle radius and the size of the target (explained later on)

Finally, the two factors used in this experiment are:

- A set of id (in our case: size 2)
- A set of distortion id (dId, in our case: size 13)

The number of trial per block is computed as: number of repetition \* size of id set \* size of dId set.

In the ControlScript, we compute each possible couple of id and dId, and store it in a list which is then shuffled. At the beginning of each trial we fetch a couple from the list and we compute:

the new target radius:

$$tR = \frac{1}{2} \cdot \frac{\text{distance}}{2^{id} - 1}$$

the new virtual target radius:

$$vTR = \frac{1}{2} \cdot \frac{\text{distance}}{2^{id+dId} - 1}$$

Where distance represents the distance between two opposite target (value set by the operator). These values are used in the experiment to display the target of the right size, and to compute the value of the distortion. Concretely, this creates 2 different displayed target sizes (radius = 0.0083m and radius = 0.0040m) and 13 virtual target radii per displayed target size. This last aspect has introduced an error in the ControlScript before the experiment: the script created 16 vTR instead of 13, resulting in unbalanced data, i.e. different number of trials per vTR. The collected data are nevertheless meaningful in regard of the perception study, and the script has been corrected.

The procedure itself of this experiment consists of:

- 1) filling a consent form from the subject
- 2) filling a characterisation questionnaire from the subject
- 3) doing a small block of 5 trial of training, to allow the subject to accommodate to the equipment and become familiar with the task
- 4) performing two block of 78 trials during each, for approximately 25 minutes per block, with a pause in between.

In the end, this experiment had 6 subject<sup>2</sup>, each completing 2 blocks of 78 trials each, for a total of 936 completed trials. This report only analyses the perception of the redirecting distortion even though, as showed later, the scripts record enough data to study throughput, precision and more.

---

<sup>2</sup> cf Appendix b) - c)

## 3.5 - Measurements

During this experiment, two different scripts are recording data.

A first script record position and rotation at each frame for selected GameObject and all collisions happening on these object. In the other hand, the Control Script logs the data needed for immediate analysis. Q1 and Q2 refer to the questions asked at the end of each trial:

- Did the virtual hand moved exactly like you ?
- Did you miss any target ?

The first question allow the subject to express his perception of the distortion in a simple way. It was designed to study perception in this work.

The latter one is meant to represent the subject awareness of the success (or not) of his reaching. It was designed to study precision perception.

The two logs implemented directly inside ControlScript are designed as follow:

Sample 1

Block nb	Trial nb	Start time	End time	tR	vTR	Q1	Q2
0	0	157.5537	168.4603	0.004032258	0.000984252	NO	YES

This log file is filled at the end of each trial and is used to compute the perception results, and could be used to measure the speed of the subject in function of the distortion strength. Unfortunately, it seems that the speed is mainly defined by the trial number. In other words, the speed of the subject increase strongly in function of time as he is getting used to the task, so much that it seems to overwrite any possible speed effect of the randomly chosen distortion strength.

Sample 2:

Block nb	Trial nb	Target triggered	Timestamp	target x	target y	target z	hand x	hand y	hand z
0	0	1	158.7333	-0.1331367	1.061526	-0.1805082	-0.1331367	1.061526	-0.1805082

This log file is filed at each target hit and could be used to measure precision when applying various degree of positive and negative distortion.

## 3.6 - Hardware precision

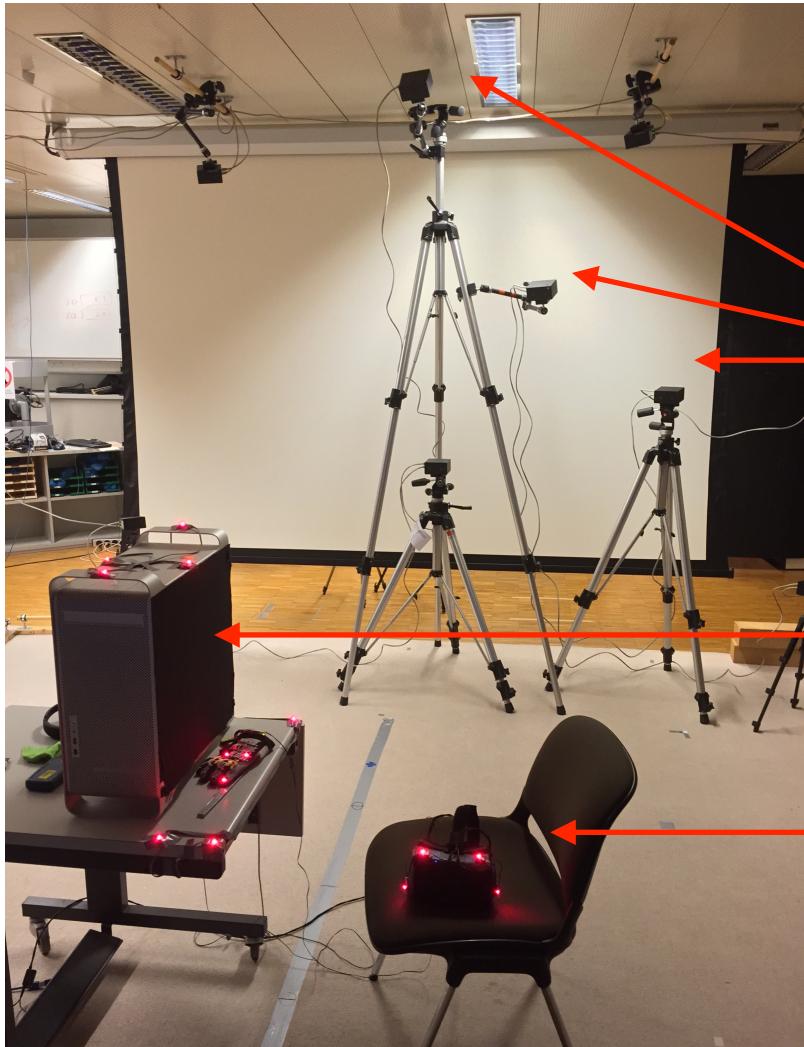


fig. 3.9 - PhaseSpace Equipment

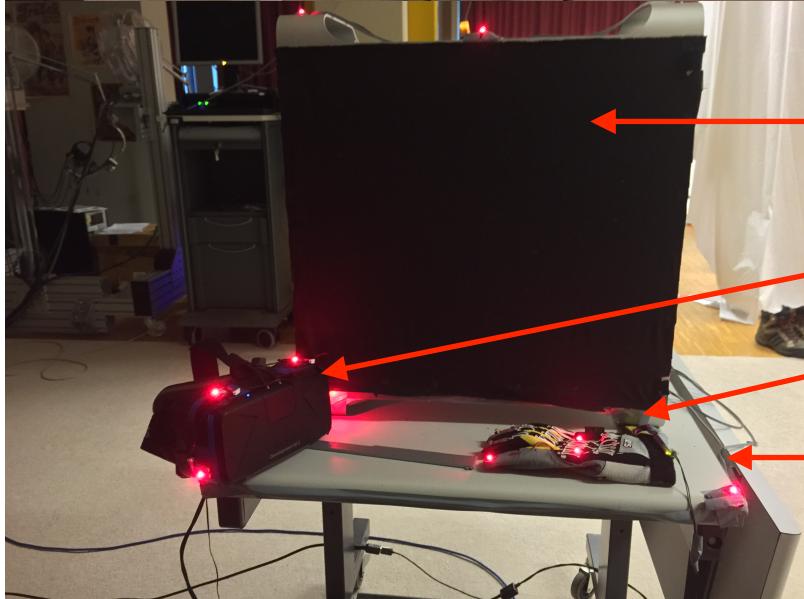


fig. 3.10 - Subject equipment

For this experiment it was necessary to attain the maximum level of measurement precision possible regarding the position of the hand. Therefore, the lab scene has been equipped with 18 PhaseSpace cameras recording at 240 Hz, strategically placed around the subject. In particular, 9 cameras following the glove and the head mounted display, the two most crucial objects. Delay is also a factor that has been taken into account. An important delay between the real world movement and the redirected movement rendered then displayed in the HMD could have a great impact on the results of the experiment. To ensure that the response time was correct, we measured it by using an iPhone 6 camera recording at 240fps. The measurement process was simple: the camera records a random led marker as well as the HMD displaying the scene from an angle where the rendered marker is visible. By turning the marker off then back on we were able to count the frames between the marker event and the rendering consequence. Even though it took more time when turning the marker from off to on, probably because the system need some time to find the marker again, total delay from marker event to displayed consequence was measured to be 50 ms.

Regarding the precision of the position, the PhaseSpace documentation claims a precision up to the millimetre.

# 4 - Results and Discussions

This study is focused on the perception thresholds and on the possible orthogonality of the two factors  $id$  and  $dId$ . We can hypothesise that the perception threshold will be greater for a positive helping distortion than for a negative one, and that the two factors  $id$  and  $dId$  will act like one single factor. We define the threshold  $t$  as  $\text{perception}(t) = 0.5$ .

The first plot (fig. 4.1) represent the perception average in function of the percentage  $(vTR/tR)^*100$ , i.e. how bigger is the virtual target radius from the target radius. Below 100%, the virtual target radius is smaller than the real one and therefore the distortion is negative, and vice-versa.

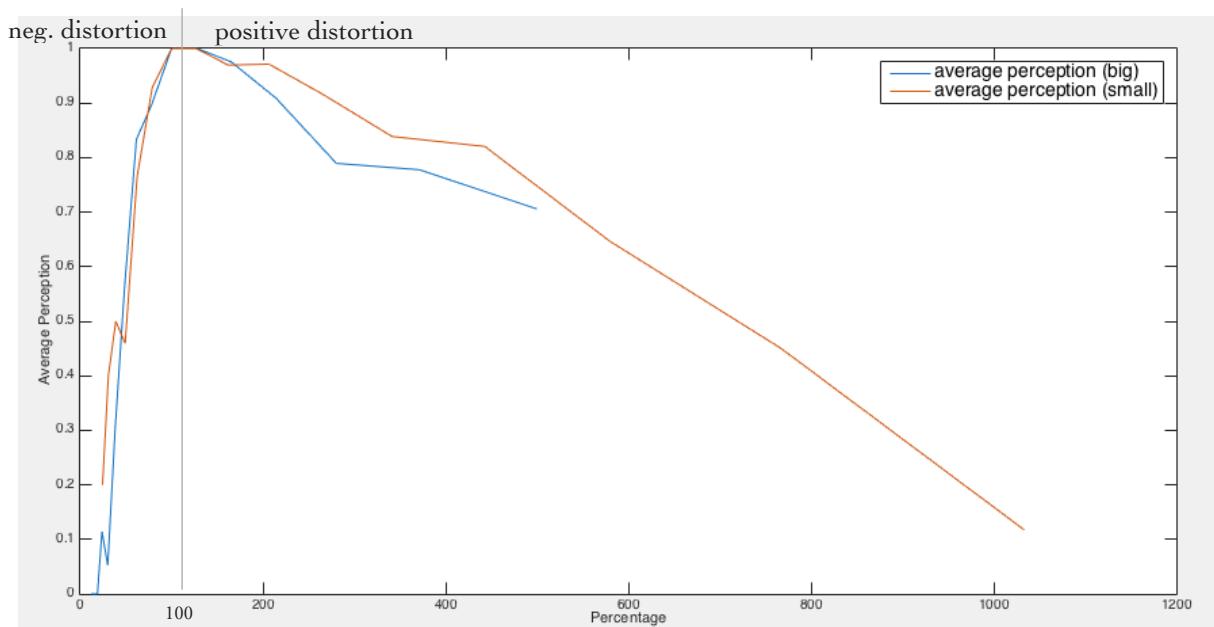


fig. 4.1 - perception in function of  $vTR/tR$

The first conclusion that comes to mind is that the perception threshold seems to be strongly higher for a positive distortion than for a negative one.

By isolating the positive distortion for the small target (fig. 4.2), we see that indeed the perception threshold for these parameter seems to be around 680%, even though the standard deviation is really strong around these values. For a positive distortion with the large target (fig. 4.3), the perception threshold was not even attained with the experiment parameters.

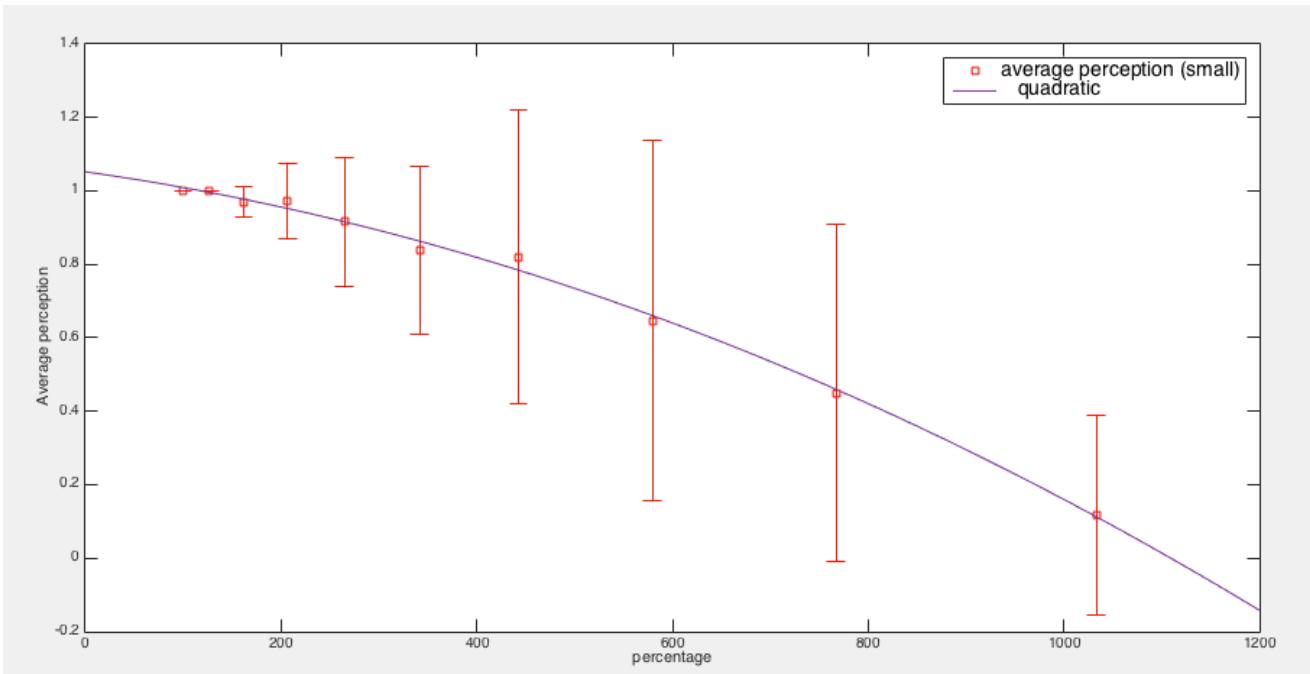


fig. 4.2 - perception averages with std deviation for positive distortion and small target size

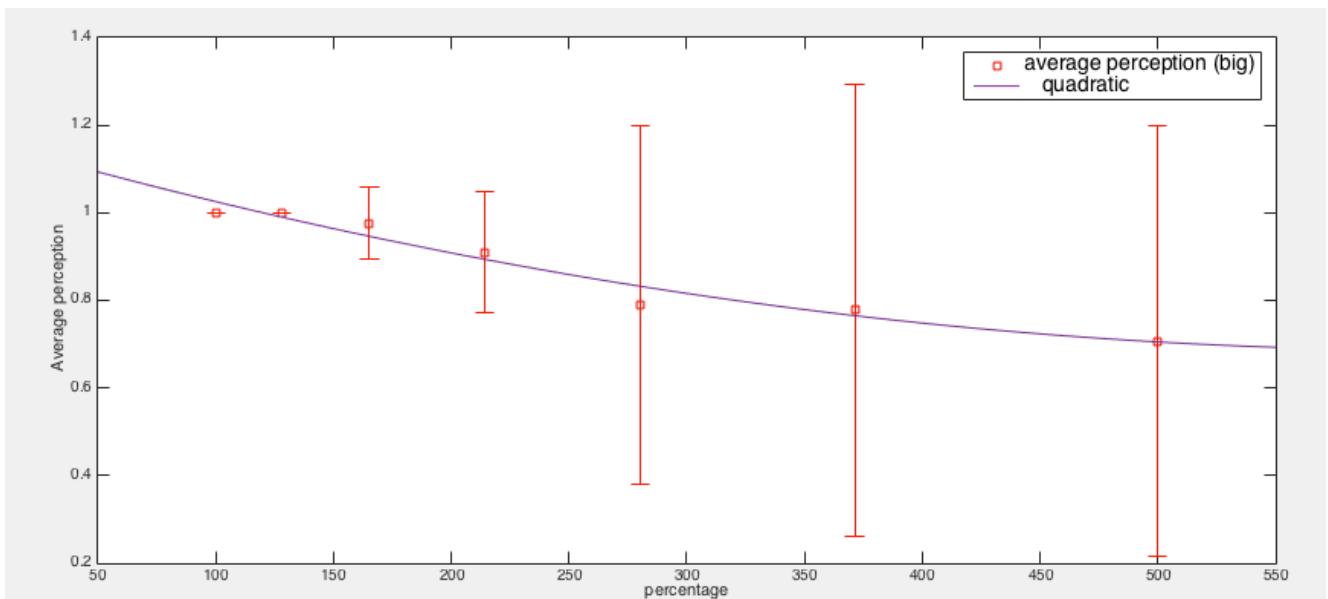


fig. 4.3 - perception averages with std deviation for positive distortion and big target size

However, the results are a lot clearer with the negative distortion. With the small targets displayed, the perception threshold seems to be at 45% when looking at quadratic extrapolations (fig. 4.4). With the big targets displayed, the perception threshold seems to be at 50%. (fig 4.5)

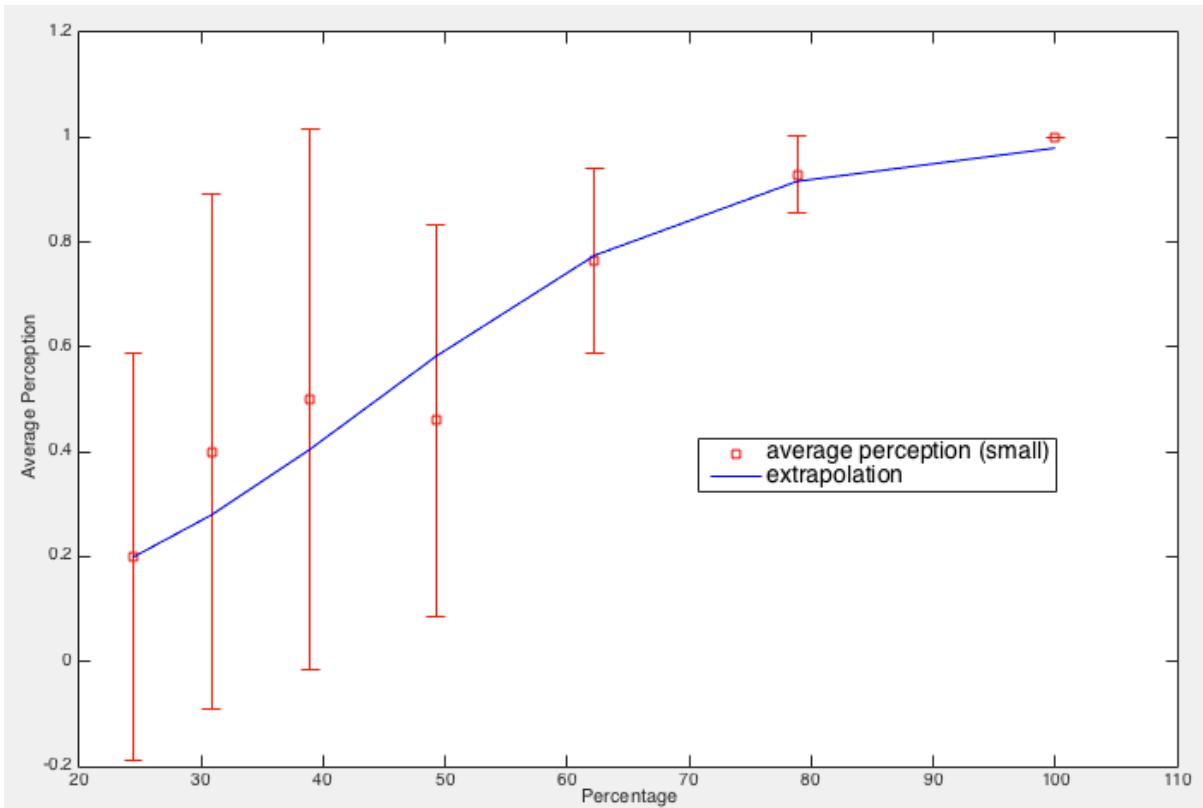


fig. 4.4 - perception averages with std deviation for negative distortion and small targets

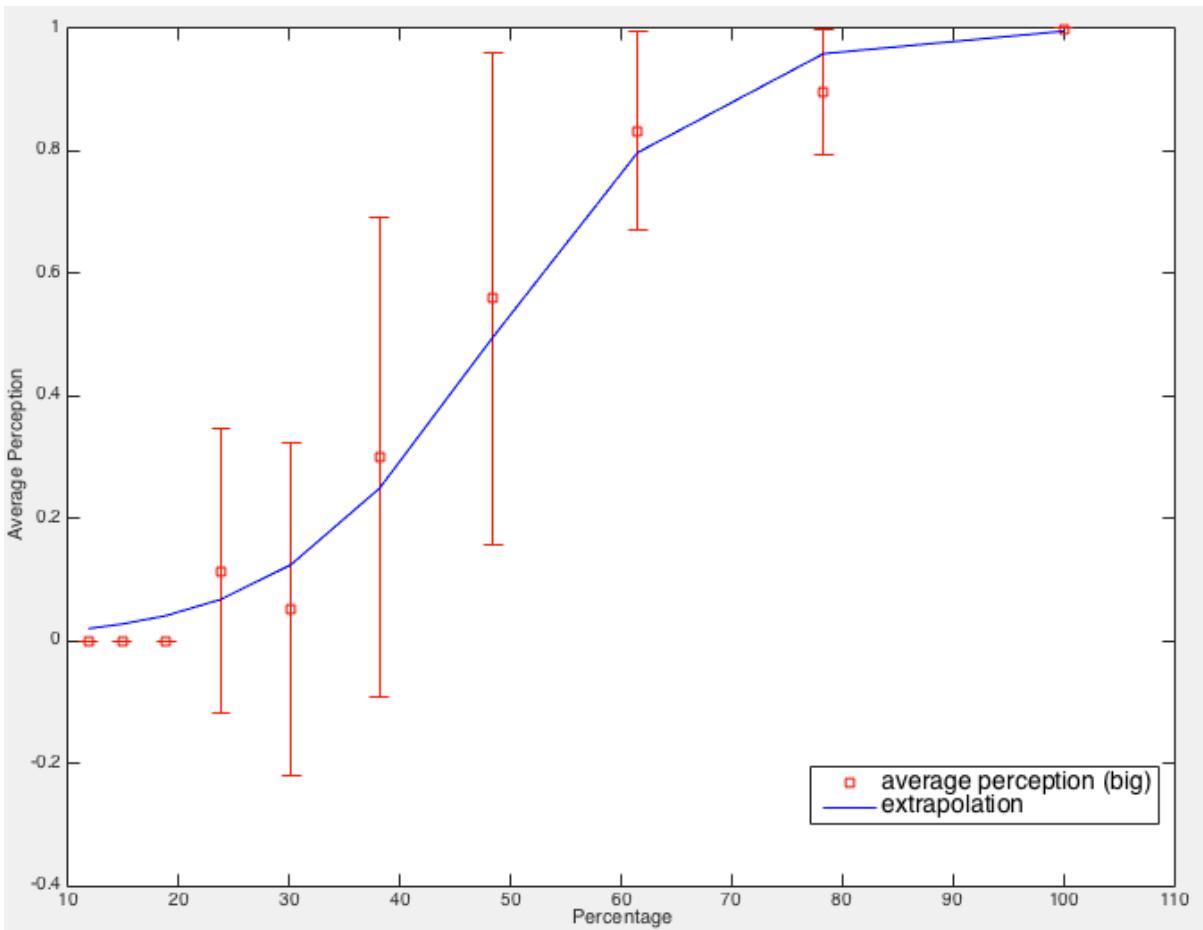


fig. 4.5 - perception averages with std deviation for negative distortion and big targets

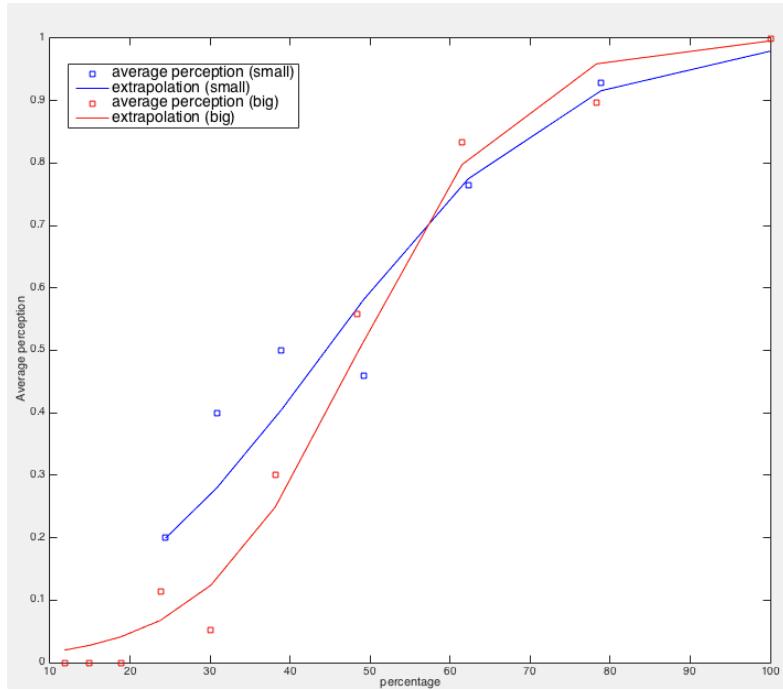


fig 4.6 - Negative distortion with both target sizes

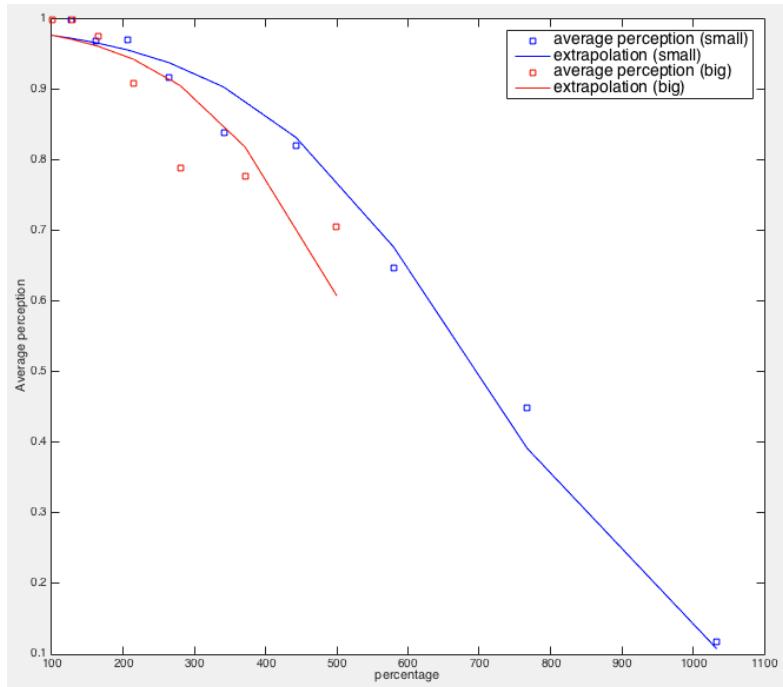


fig 4.7 - Positive distortion with both target sizes

Regarding the thresholds themselves, the first conclusion we can make is that all the subjects with no exception were a lot more sensible to the negative distortion than to the positive one. If we consider the above-estimated thresholds, we can compute

$$\text{perceptionMin+} = |100\% - \text{positive\_threshold}| = |100 - 680| = 580\%$$

$$\text{perceptionMin-} = |100\% - \text{negative\_threshold}| = |100 - 45| = 55\%$$

These distances represent the average strength of redirection needed to be applied to the subject hand to be perceived.

When overlapping both targets size for the negative and the positive distortion on a plot, (fig 4.6, 4.7), it seems that the two factors are not parallel as thought. But this may be caused by an insufficient number of experiment data causing the results to be sparse. It is possible that, when adding more and more subject answers, the perceptions average tends to form curves similar enough between small and big target displayed, to consider both these parameters as parallel, i.e. parameters that can be treated as one.

# 5 - Conclusion

As the results show, the data collected with 6 subject and with wrongly set parameters resulted in unbalanced data. However, it sets the ground for further experimentation and analysis, and implements all required tools, scripts, scenes and log needed to give to the IIG lab the possibility to study various effects of the redirecting distortion. Speed, throughput, precision, perception of the precision, 3D rendering of the subject movements ...

For now, we can conclude that humans tends to be extremely sensible to modification of their movements that make their task harder, but they are a lot more tolerant to a helping redirection. Both of these aspect could have various real-world application. For example, the negative distortion aiming at making one's accuracy around a target, could be used to train precision. For instance, a surgeon could train his precision by seeing all of his little residual movement amplified around the most sensible part of the operation. In the other hand, the positive redirection could be used in mass market virtual-reality. Indeed, the day VR will be accessible from any computer or gaming system is really close, and the need to make reaching task easier will be present from the beginning. As assisted aiming helps novice gamers to target enemies in shooting video-games, the positive redirecting could be used to help the VR user to reach and grab an object, or activate a virtual button. It could be implemented to reduce the effort needed to complete a task while requiring a smaller tracking space ... By activating the distortion when entering a certain range, and by making this distortion stronger while the hand becomes closer, the redirecting could be an implementation model to predict and help the user to perform his intended movement. And this redirecting is nothing less than the exact distortion computed here.

# 6 - References

Marc Jeannerod, "The mechanism of self-recognition in humans", *Behavioural Brain Research* 142 (2003) 1–15

O.A. Kannapea, L. Schwabea, T. Tadia, O. Blankea, "The limits of agency in walking humans", *Neuropsychologia* (2010)

Frank Steinicke et al., "Estimation of Detection Thresholds for Redirected Walking Techniques", *IEEE transaction on visualization and computer graphics*, vol. 16, n°1, 2010

Luv Kohli, Mary C. Whitton, Frederick P. Brooks Jr., "Redirected Touching: The Effect of Warping Space on Task Performance", *The University of North Carolina at Chapel Hill* (2012)

Luv Kohli, Mary C. Whitton, Frederick P. Brooks Jr., "Redirected Touching: Training and Adaptation in Warped Virtual Spaces", *The University of North Carolina at Chapel Hill* (2013)

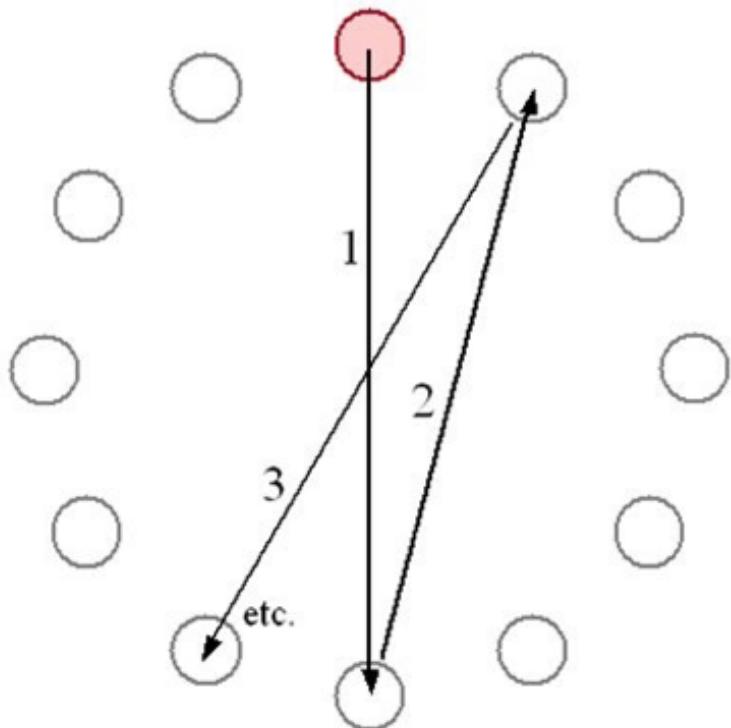
ISO/TS 9241-411:2012(E) Testing of efficiency and effectiveness (*Annex B*, 2012)

# 7 - Appendix

## a) Experiment Procedure<sup>3</sup>:

Make sure your cellphone is off, and will not disturb you during the experiment.

You will be touching all the targets in a circle several times



Your movement may be replicated perfectly, or redirected in order to increase or reduce the difficulty of the task. You must not deliberately move your hand in order to identify such redirections, you must instead do the aforementioned task as quickly and accurately as you can, avoiding any additional movement. Make sure to raise your finger over the surface before moving.

For each round of reaching you will answer 2 yes/no questions:

Did the virtual hand moved exactly like you?

Did you missed any target?

This experiment might be straining to your right arm, use the interval between answering the second question and reaching the first target of the next round to rest.

Tell the experimenter whenever you need a break during the experiment. He will help you removing the equipment. If possible, ask for the break during the interval between answering the second question and reaching the first target of the next reaching round.

<sup>3</sup> Image from <http://www.yorku.ca/mack/nordichi2012.html>

## b) Characterisation questionnaire

08/06/15 06:50

Pre experiment

\*Obligatoire

**1. Identifier (filled by the experimenter) \***

.....

**2. Age \***

.....

**3. Length of index finger \***

.....

**4. Gender \***

*Une seule réponse possible.*

Male

Female

**5. How often do you participate on experiments using Virtual Reality equipments? \***

e.g. experiments in other labs of the university.

*Une seule réponse possible.*

Never participated of an experiment

A few times

Every month

Every week

Every day

**6. How often do you use head mounted displays? \***

*Une seule réponse possible.*

Never used

A few times

Every month

Every week

Every day

**7. How often do you play video games? \****Une seule réponse possible.*

- Never played
- A few times
- Every month
- Every week
- Every day

**8. How often do use the Microsoft Kinect, Nintendo Wii or Playstation move? \****Une seule réponse possible.*

- Never used
- A few times
- Every month
- Every week
- Every day

**9. Hand of preference \***

usually, the hand you write with

*Une seule réponse possible.*

- Left hand
- Right hand

**10. Area(s) of expertise/study/work/interest \***e.g. computer science, math, sales,  
mechanical engineering etc.**11. Are you a student? \****Une seule réponse possible.*

- Yes, bachelor student
- Yes, master student
- Yes, PhD student
- No, I'm not a student

Fourni par



### c) Characterisation results

id	Age	Gender	Participate on exp. using VR equipment s?	Use HMD ?	Play video games ?	Use Kinect, Wii or PS move?	Dominant hand	Area(s)	Student?	Length of index
1	22	F	A few times	Never used	Never played	Never used	Right hand	math	Yes, bachelor student	7
2	19	M	A few times	A few times	Every week	A few times	Right hand	civil engineering	Yes, bachelor student	8
3	25	F	Never participated of an experiment	Never used	A few times	A few times	Right hand	international cooperation	No, I'm not a student	7
4	25	F	A few times	A few times	A few times	A few times	Right hand	environmental engineering	Yes, master student	6.5
5	26	M	Never participated of an experiment	Never used	A few times	Never used	Right hand	Applied math and Computer	Yes, PhD student	7.5
6	26	M	A few times	A few times	Every month	A few times	Right hand	civil engineering	Yes, master student	7.5