Introduction

When designing a completely new software system there are many factors to consider but the most important ones are the future end users. This is especially true for systems aimed a narrower group of end users as it will provide the possibility to tailor make the software to a much greater extent. Depending on the background of the end user restrictions can be put on the software system and demands might vary greatly. As a result of this involving the user in the actual software development process can produce a superior software system, the option to involve the user in the entire process might not always be available but if it is the user can prove to be a invaluable asset in the software development process.

Motion vision is very important for any seeing species in the animal kingdom, for example how animals track moving targets in their field of vision or how they generate an optic flow as they themselves move to be able to go in the desired direction (Borst and Euler 2011). There are several aspects of motion vision to study, an important one is to study at behavior as a function of visual stimuli.

Using insects for doing behavioral analysis is a common technique as studying vertebrates is a complex endeavor and surprisingly enough certain insects code motion vision in a fairly similar manner as vertebrates do.

Getting a better understanding of motion vision is important for advanced technology such as self driving cars, air planes and robots. Learning how motion vision is processed in for example insects is particularly interesting as they apparently process this rapidly despite having very limited brain capacity.

2 Background

* 1. Motion vision at Uppsala University

At the motion vision lab in the neuroscience department of Uppsala University fly's, and in particular hoverfly's, are used for their research. For doing behavioral research different methods are used, for example you can record the fly's movement in a contained area while presenting visual stimuli for it and then do image analysis of the recorded material to get the necessary data. Another technique that the lab at Uppsala University is using is a trackball setup on which a tethered fly can walk on to generate a virtual movement path.

The setup is built by using two optical sensors extracted from two high quality gaming mice. These are held in place in such a way that they are perpendicular to each other and aimed at the center of the cup where the ball is placed. Using light airflow streaming from the bottom of the cup the ball is hovering slightly and very easy to spin which is necessary for the fly to be able to rotate the ball. The fly itself is then tethered to a tube so that it hangs low enough to be able to walk on the ball and high enough not to be pushed into it. Furthermore the fly is aimed at a CRT monitor which in the experiment is going to be used for showing visual stimuli.

Insert figures of setup here  
  
The two mouse sensors on the trackball setup are going to be used as simple motion detectors. This means that there is a need for software that can read the raw mouse data and decode these to the fly’s delta coordinates. As there are little to none ready-made and available software alternatives for this purpose a new software system is needed.

* 1. Agile software development

Agile software development methods are the group of methods that follow the principles of the agile manifesto. They are also considered to be the opposite of the other large group of software development methods, i.e. plan based processes.(Beyer) In plan-based methods all activities needs to be planned and scheduled extensively before they are started.(Sommerville) In contrast to this, trademarks of the agile methodology are rapid design, development and releases of working prototypes, welcoming changes in customer requirements rather than the opposite and a lot of informal face-to-face communication as opposed to communicating through large amounts of unnecessary documentation. (Paulk 2002) Agile methods were developed as a result of plan based methods had poor performance. Large project were planned ahead of time and during the project requirements changed and time schedules were to tight which in the end lead to abandoned projects. (Beyer)

In many of the agile methods, for example Scrum, work is divided into shorter sprints. Sprints are initialized by choosing requirements to implement during that sprint, these requirements are then frozen for that time and when the sprint is over these should be fully implemented and testable. This leads to the end of the sprint where an evaluation session is held where the customer can test the working prototype and give feedback on the project. During this session there is potential for adding, removing or updating requirements along with changing customer demands. (Beyer)

* 1. Adding user-centered practices

User-centered software design can have different focus and be using different tools depending on the work environment and what the type of software system to be implemented. Some of the tools described in my project are described below.

Before the analysis and design phase can begin however, the initial phase of a user-centered systems design is launched. This is called “phase 0” and contains the overall project description. (Beyer)

**2.3.1 Contextual inquiry**

Contextual inquiry means that the design team perform an analysis of the end-user by observing them doing their actual work and seeing in which manner they perform these tasks is what is known as a contextual interview. In this phase, design of the system is not in focus but rather how the working environment looks like and how work generally is done in this particular context. The objective is not so much as looking for problems but rather to understand the users and their need.*[[1]](#footnote-1)* After the contextual inquiry all notes and data gathered about the user needs to be organized in a logical fashion. The data might be overwhelming though and it can be difficult to structure, therefore an *interpretation session* is held. *Affinity diagrams* are a way of organizing the data. These diagrams are built bottom up by building small groups of items that have a connection and then iteratively build larger groups of the newly created groups.[[2]](#footnote-2)

**2.3.2 Sequence models**

*A sequences model* is a model describing user strategies for performing tasks and define the intent and the steps that are necessary to perform to finish the task. The model describes in detail exactly what the user currently do when solving certain problems. Modeling this is to get an overview of what is actually happening and where issues might lie. A sequence model is therefore a good tool to possibly find better strategies than the ones currently employed by the users.[[3]](#footnote-3)

**2.3.3 Scenarios**

There are different types of usages for scenarios, one of them are conceptual scenarios. A conceptual scenario is an abstract description of how a user performs a specific task, there is no detailed description of tools and technologies are used or suggestions for how the tasks should be solved. Using conceptual scenarios is a good way to generate requirements and it will lead into concrete scenarios which are descriptions with much higher level of details and these will start provide suggestions for the future interface design (Benyon 2010).

**2.3.4 Prototypes**

Using prototypes is a good way to early on in the design phase provides the user with a clear description of how the interface is going to look like. There are also ways of letting the user try these out whether the designer is using low-tech prototypes such as paper prototypes or via some prototyping software creating an interactive interface without its actual functionalities. If the case is the former the designer can use post-it notes as pop-ups and covering invisible parts with paper and removing it as the user makes them visible by its interacting with it. Paper prototypes are a good tool for getting the user to get a sense of the look and feel of the future system without actually having to program a GUI. This makes it easier to do major changes to the structure of the GUI as you will not have to program it. As the project moves on these prototypes won't be sufficient but as the actual GUI is produced the user can give feedback on that instead and hopefully any future changes will not have to be major.[[4]](#footnote-4)

* 1. Usability

Usability is a very important concept of human-computer interaction and is a measurement you can use in an evaluation of a system’s user friendliness. What is meant by usability is that usage of the system should require as little effort as necessary when performing the tasks needed. The system should provide the user with necessary information and that it is organized in a natural way in the graphical user interface. That the software has a short learning curve is also a feature of a system with high usability.[[5]](#footnote-5)

Design principles are an important tool for shortening the learning curve of the system for naive users. Design principles are artifacts that are often found in similar systems that are easy to recognize for the users. This means simple artifacts such as “undo”-/”redo”-buttons, “back”-buttons as a way of navigating and generally just placing interface items where they normally would be in a similar system.[[6]](#footnote-6)

Emphasis on usability tends to diminish in the software process for two reasons in particular. One reason is due to the fact that the customer rarely specify the fact that they want a usable software system (they think it's implied) and when the company delivers the software it is not as user friendly as the customer might have wanted. This is because the customer probably isn’t a HCI expert and might not even know about the term usability although on an abstract level they want a system that is easy to use but never realize they need to emphasize this. Therefore this is something that is important for the designers to bring up and discuss with their clients. Another issue that might arise is that usability experts are normally only actively participating in the early parts of the software development process and not as much in the implementation and testing phase. This is mainly because usability experts rarely are software developers. What is needed to solve this is HCI experts that are familiar enough with programming to be able to participate actively in the implementation phase in particular but also in the validation phase where system requirements are validated (including usability requirements).[[7]](#footnote-7)

* 1. Evaluating usability

Beror lite på hur slutliga utvärderingen blir, skriver därför detta i samband med utvärderingen.

1. Purpose and methodology
   1. Purpose

The motion vision lab needs a brand new user friendly system that can perform the tasks needed to study behavior of fly’s using a track ball setup as a hardware solution. To verify that the goals of high usability have been evaluation of the finished software system needed to be done.

* 1. Choice of development process

As I as a developer and designer worked in the very same lab as the end users in the motion vision lab at Uppsala University choosing an agile development process was natural. The initial phase of the project, also called phase 0, was devoted to project description. This includes finding and analyzing requirements of the software system, design a suitable graphical user interface and planning the implementation phase.

The implementation phase was divided into four sprints of two weeks each. Each sprint was finalized by having a walkthrough of the progress so far where FlyTracker was tested by checking of the list of requirement for that particular sprint. This made it possible to every second week get very valuable feedback and to make sure that the development was on the right track. Furthermore, this also made it important to divide the work such that functions (or requirements) were completely implemented during its respective sprint. In the middle of every sprint there was also a shorter check-up meeting where we discussed potential problems or changes to the requirements before they were fully implemented.

Working according to the principles of user centered design was also a clear choice given that the end users are few and that I am working in the same lab as they are. This meant that there in addition to the scheduled meeting were a lot of informal discussions about the project and questions could be asked about even the smallest details as they were implemented. This is something that the agile manifesto is clearly emphasizing.

* 1. Implementation

The employees in the motion vision lab at Uppsala University are familiar with Matlab and also have some experience with Python. Therefore there was a requirement that the software system was implemented in Matlab. Using Python for certain modules of the system was acceptable as it was mandatory for a few specific functions.

* 1. Evaluation

The evaluation was performed by letting the main user of FlyTracker try it out for a few weeks. An interview was then conducted where she got to answer question about (see appendix xx) the system and the results of this interview were analyzed and no major problems with usability were found. Skriver klart när jag vet exakt hur det går till

1. Beyer, Hugh, *User-Centered Agile Methods* [↑](#footnote-ref-1)
2. Beyer, Hugh, *User-Centered Agile Methods* [↑](#footnote-ref-2)
3. Beyer, Hugh, *User-Centered Agile Methods* [↑](#footnote-ref-3)
4. Benyon, s.184-187 [↑](#footnote-ref-4)
5. Benyon, David, *”Designing Interactive Systems”,* s. 84 [↑](#footnote-ref-5)
6. Benyon, D s.89 [↑](#footnote-ref-6)
7. Göransson, B, Gulliksen, J, Boivie, I, *”The Usability Design Process – Integrating User-Centered Systems Design in the Software Development Process”* [↑](#footnote-ref-7)