Acquisition of Expanding Targets

Summary

Svetlana Slavova
sds797@mail.usask.ca

ABSTRACT

This work is a summary of the paper, presented in [1].

INTRODUCTION

Expanding targets is a dynamic interaction technique that is used to facilitate target selection, taking into account the current focus of the user. This approach allows more icons to be displayed in a reduced size, and particular items to be enlarged, only when it is necessary.

The Fitts' law, valid for static icons, specifies that if the width of the target is decreased, more time is needed for an item selection. However, the law does not indicate how the selection time is changed, in case of dynamically expanding targets.

PROBLEM DEFINITION

In order to determine whether the initial size and/or the final size of the items influence the selection time, as well as if the Fitts' law can be applied in expanding targets, the dynamic interaction technique has to be evaluated. The results of such an investigation will give an insight how to build interfaces, in order to improve performance, as well as what kind of issues may arise.

APPROACH

The authors of [1] rely on the Fitts' law, which gives the movement time, needed to select a target with a particular width and distance from the user. The "optimal initial impulse model" is considered. It suggests that the target acquisition movement consists of a main sub-movement and a set of corrective fast sub-movements.

The authors propose that the selection time of expanding targets should be better than the selection time of the static ones. In addition, it should depend on the final size of the target, rather than on the original size, and should be predictable by the Fitts' law.

EVALUATION

Experiment setup

The conducted experiment consists of two items – a start position of the mouse cursor and a target. Two designs are taken into account: a static design – no expansion is considered; and a spatial expansion – the width of the target starts increasing gradually to a maximum width value, when the cursor reaches a particular distance from the target, moving towards the target. Different indices of difficulty are taken into account too. The task is to move the cursor horizontally from the start position towards the target as fast as possible.

Twelve users, divided into two groups, participated in the experiment. Each group started the experiment with a different design, and then switched the designs.

Results

As expected, the movement times for special expansion are lower than the times for the static design. In addition, the selection time of dynamically expanding targets depends on the final size of the item, rather than on its original size. Furthermore, the performance can be calculated, using the Fitts' law for static targets. The obtained results indicate that the dynamically expanding targets technique can improve performance in a predictable manner.

DESIGN ISSUES

The results of the experiment lead to several design issues, which should be taken into account when building interfaces with expanding targets. In order to define which item or set of items to expand, the position of the cursor can be taken into account. Thus, the icons that are close to the cursor are enlarged. However, this may cause items overlapping or occlusion. In addition, the expansion may distract the user's focus or make the selection harder, if the items are moved away significantly from their original position.

CONCLUSIONS

The paper, presented in [1], verifies that expanding targets in an applicable technique for improving performance, which can be estimated by the Fitts' law, applied for the static case. In addition, the evaluation confirms that the selection time depends on the final size of the target, rather than on the initial one.

REFERENCES

1. McGuffin, M., Balakrishnan, R. Acquisition of Expanding Targets. *CHI'02*, Minneapolis, USA, 20-25 April, 2002.

Acquisition of Expanding Targets

Implementation Description

Svetlana Slavova
sds797@mail.usask.ca

ABSTRACT

This paper is a description of my implementation of the expanding targets interaction technique, presented in [1].

DESCRIPTION

The program represents an implementation of a target expansion interaction technique. The environment consists of two main objects, displayed as white boxes – a start position of the mouse cursor and an expanding target. The target appears when the cursor is located to its starting position. The cursor has to be moved horizontally towards the target. The target object starts expanding gradually from its original width to its maximum width (1.5 times the target's initial width), when a certain distance from the target is reached. The expansion effect can be observed only between the start position box and the target object. In order to visualize better the expansion level, two green lines show the initial width of the target, whereas two red lines specify the maximum expansion of the target.

IMPLEMENTATION DETAILS

The program is implemented in Java, using Java Swing. Two classes are developed, as follows:

- *Expansion*. This is the starting point of the application. The class is responsible for the creation of the program's frame. It adds to the frame the mouse surface, which is represented by an object of class *MousePanel*, described below.
- *MousePanel*. This class represents the mouse surface. It handles the mouse events, reacts, according to the performed actions, and updates the user interface. The main components of the class are the following:
    - Method *drawObjects* – it displays the static objects of the user interface, such as background, start position of the mouse, and some instructions, regarding how to use the program;
    - Method *drawTarget* – it displays the target, as well as the target's initial width and the target's maximum width are drawn in green and red, respectively;
    - Class *MyMouseMotionAdapter* – it handles the movements of the mouse. Every time when the mouse is moved, method *mouseMoved* of the class is called. It updates the mouse coordinates and invokes method *paintComponent*, which repaints the user interface;
    - Method *paintComponent* – it contains the main logic of the program. It is called when the user interface has to be repainted. The previous and the current location of the mouse are taken into account, in order to calculate whether or not to draw the target, and what width of the target should be drawn. The algorithm of the method is presented on the diagram below.

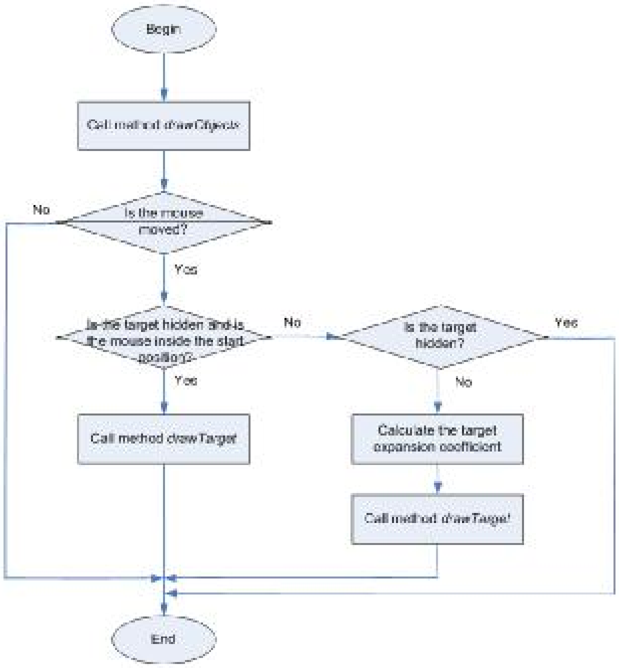


Figure 1. Algorithm of method *paintComponent*

HOW TO COMPILE & RUN

- In order to compile the program, type the following command in the terminal window:

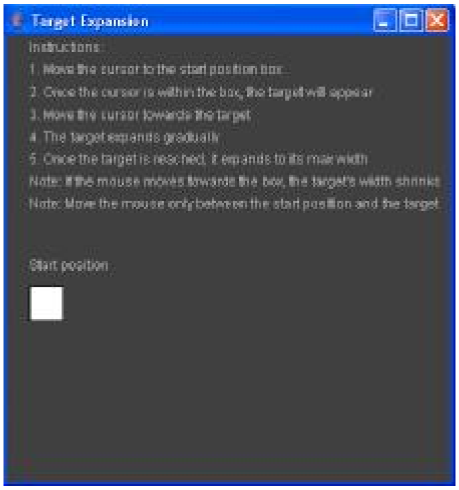
  javac Expansion.java

- In order to run the program, type:

  java Expansion

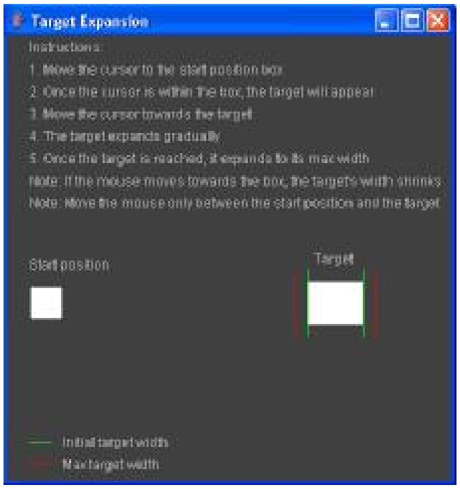
USER INTERFACE

When the application is started, a window, shown on figure 2, is displayed. Once the mouse is located in the start position box, the target appears in its initial size, as shown on figure 3. Figure 4 shows how the target expands, when the mouse cursor moves towards it. Finally, figure 5 shows the fully expanded target.



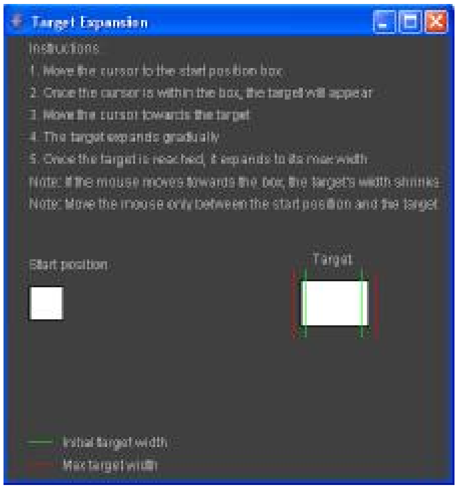
Instructions
1. Move the cursor to the start position box
2. Once the cursor is within the box, the target will appear
3. Move the cursor towards the target
4. The target expands gradually
5. Once the target is reached, it expands to its max width
Note: If the mouse moves towards the box, the target's width shrinks
Note: Move the mouse only between the start position and the target

Figure 2. Initial window of the application



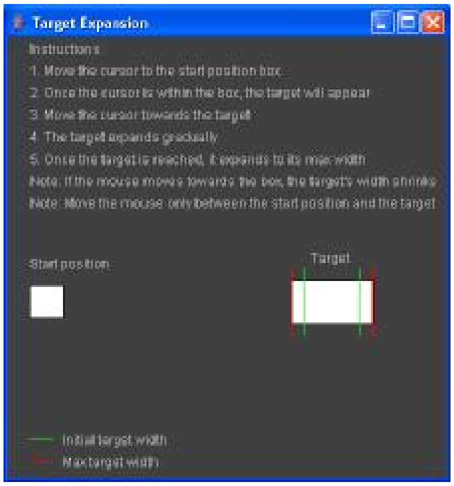
Instructions
1. Move the cursor to the start position box
2. Once the cursor is within the box, the target will appear
3. Move the cursor towards the target
4. The target expands gradually
5. Once the target is reached, it expands to its max width
Note: If the mouse moves towards the box, the target's width shrinks
Note: Move the mouse only between the start position and the target

Figure 3. The target is displayed, when the mouse cursor is located in the start position box



Instructions
1. Move the cursor to the start position box
2. Once the cursor is within the box, the target will appear
3. Move the cursor towards the target
4. The target expands gradually
5. Once the target is reached, it expands to its max width
Note: If the mouse moves towards the box, the target's width shrinks
Note: Move the mouse only between the start position and the target

Figure 4. Partially expanded target



Instructions
1. Move the cursor to the start position box
2. Once the cursor is within the box, the target will appear
3. Move the cursor towards the target
4. The target expands gradually
5. Once the target is reached, it expands to its max width
Note: If the mouse moves towards the box, the target's width shrinks
Note: Move the mouse only between the start position and the target

Figure 5. Fully expanded target

REFERENCES

1. McGuffin, M., Balakrishnan, R. Acquisition of Expanding Targets. *CHI'02*, Minneapolis, USA, 20-25 April, 2002.