University of Edinburgh

School of Informatics

HCI Image Labelling Application

Sam Shelley (s1146283) and Khurram Aslam (s0792198)

October 21, 2011

**1. Introduction**

The Image Labeller application facilitates the completion of a repetitive task: identifying and labeling objects in images. The key focus throughout the development of this enhanced user interface is therefore reducing the number of tasks necessary to identify and label an image.

Alan Dix, a Human-Computer Interaction expert, consolidates good usability principles into three main categories; Learnability, Flexibility and Robustness. Using the principles in each grouping as a guideline, we were able to determine the best way to improve the user interface and ultimately the user experience of the Image Labelling application.

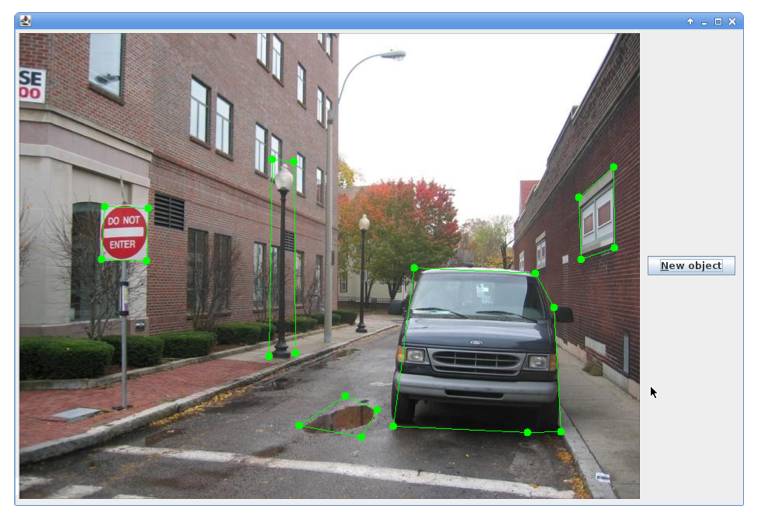
In the following report we will briefly cover each of these groups and the principles within each group to demonstrate how we applied.

**2. Learnability**

Learnability focuses on evaluating the ease with which new users can start using the application and perform tasks optimally. By adhering to these principles we were able to improve the accessibility of our interface.

**2.1 Predictability**

The basic interface provided for the Image Labeller was moderately predictable. Clicking on a point on the image will always create a point and draw a line connecting it and the previous point. Clicking the “New object” button consistently finishes the polygon. This button, however, has poor operation visibility since it is perpetually enabled, when finishing the polygon is only possible when the user has already created three or more points beforehand (A fact which is not immediately obvious to a first-time user).

One possible solution might be to disable the button unless the user has created three or more points, but this behavior is still non-intuitive since it is not clear whether you should press “New object” when you have drawn a complete polygon with no gaps, or whether you should press the button to automatically complete the gap (the intended behavior).

Furthermore, since the user will be interacting with the image to draw the polygon, his or her mouse will be pointed towards the image. The mouse will therefore be on average half of the width of the image away from the button. Fitt’s law demonstrates a positive correlation between the amount of time it takes the user to move the mouse to a required location (and thus how long it takes to tag an object) and the distance the object is from the starting point.

**Fitts Law:**  T=a+blog2(DW+1)

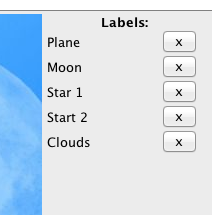
In the enhanced version of the Image Labelling application, we remove the “new object” button, instead allowing the user to complete a polygon by clicking on its starting point. The start points are much smaller than the button, so we increased their dimensions, but still reduced the clickable width to approximately ¼ when compared to the “new object” button. However, we make up for the width reduction with a dramatic reduction in distance (remember that the user must travel to the finish object button, and then back again to the image). Therefore, assuming that we start in the center, we’ve reduced traveling distance by about ¼. The real gain, therefore, is in the form of the improved predictability of actions (the user may now conduct all drawing by clicking on the image) and familiarity (which is discussed later).

In addition to removing the “new object” button, we added a menu on the top of the window which is consistently placed in one location. The “Undo” and “Redo” options function predictably, undoing or redoing any action the user has made on a given image. Additionally they provide high operation visibility by greying out when the user has no actions to Undo or Redo.

**2.2 Synthesizability**

The goal of the interface is to create polygons around objects in the image and the interface provides immediate response to indicate whether or not the user is reaching that goal (it draws the next segment and point whenever the user clicks)

When a polygon is completed, a box immediately appears prompting the user for a label (no additional clicks are necessary). This takes a decision step away from the user, who may focus only on completing the polygon, and then immediately typing in the preferred label name.



Additionally, by lining up the buttons to delete a given label, we have decreased the time it may take the user to delete a batch of labels at one time. If these buttons were instead placed at the end of each label, the user would need to move their pointer horizontally to click each button. The alignment ensures that the user only needs to move their cursor along the vertical axis to delete several labels.

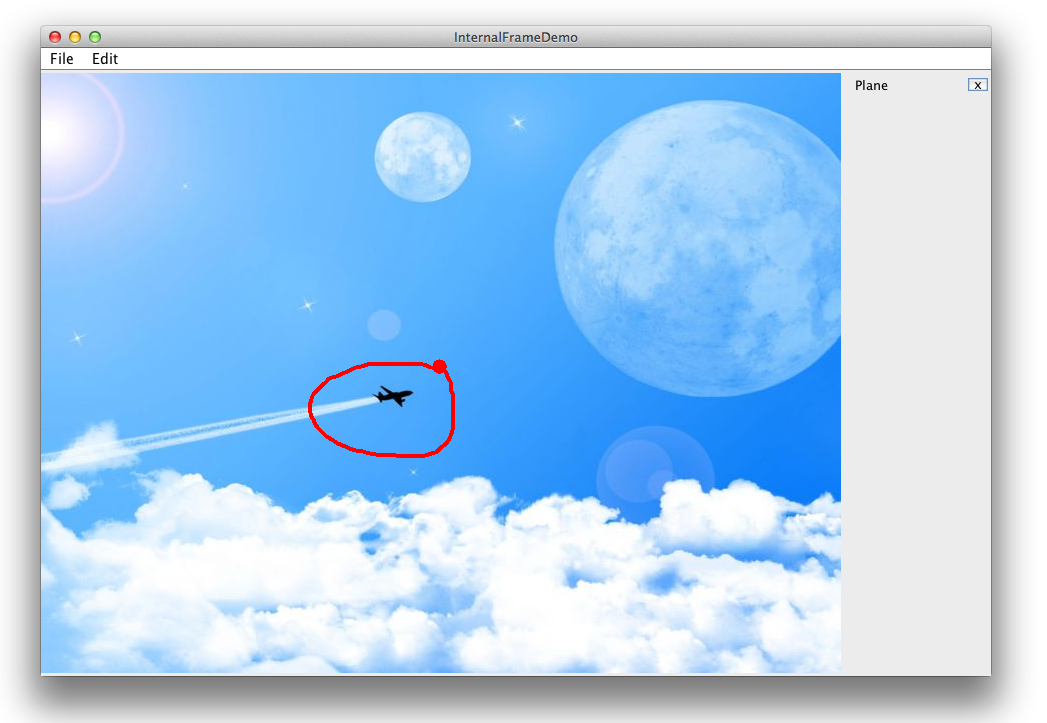
**2.3 Familiarity**

There are two types of personas who will generally be using the Image Labeller. The first peronsa could be Susan, a middle aged woman who has been asked to participate in the research project by her son who works at a University. Susan has had very little experience working with computers and approaches the interface with an almost entirely fresh perspective.

For Susan, the concept of drawing a polygon on the image will most clearly relate to drawing with a pen and pencil in the physical world. The interface should thus make *affordances* for her past knowledge. If given a piece of paper with an image and told to draw polygons around the objects on the page, Susan would draw a continuous shape around the object and end at the starting point.

Finishing a polygon by clicking the “new object” button is thus counter-intuitive. She may try completing the polygon by clicking on the start point, which is incorrect behavior in the provided application (since this will create duplicate points). Removing the “new object” button therefore improves the familiarity of the interface for Susan. Additionally, Susan might be uncomfortable with drawing with endpoints as it is very unlike real-life drawing. Freeform drawing therefore aligns more with what Susan might expect.

A second potential persona is Max, who is comfortable with basic applications on his computer. He has solid knowledge of Microsoft Word and some basic drawing application. For Max, it is important to conform to the existing standards of these applications. Paint and drawing applications typically allow the user to draw freeform shapes on the canvas.



Freeform drawing of shapes therefore also makes the application more familiar t Max

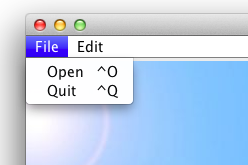
.

Max is also comfortable with the menus and shortcuts that exist in other applications on his computer. We therefore make use of expected shortcuts (CTRL+Z for undo, CTRL+Y for redo, etc...) so that the actions he is familiar with apply to this application.

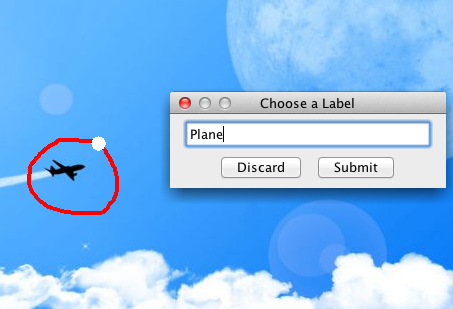
**2.4 Generalizability**

The enhanced Image Labeller incorporates a menu bar that consists of a ‘File’ and ‘Edit’ menu item. This gives the application an appearance that is typical to other applications on their existing system. The user would generally expect to be able to Open and Quit the application from the File menu, and to find the Undo and Redo functionality within the Edit menu. Thus we have conformed to these standards.

To increase the similarity between the Image Labeller and the conceptual model the user may have, we have implemented shortcut keys for all the menu items that are consistent with similar existing applications that the user may have on their system.

Users may be familiar with the concept of “Undo” and “Redo”. Within an image editing applications, Undo and Redo functionality is expected given the frequent mistakes a user makes when drawing. Therefore we have implemented these functions and made them easily accessible in the Edit menu. These actions can also be used by their corresponding shortcuts which is beneficial to experienced power users.

**2.5 Consistency**

One of the goals in our enhanced Image Labelling application is to maintain consistency throughout all actions and outcomes in the user’s work-flow (drawing and labelling images). It is very important for both power users and first-time users to correctly assume that they can perform the same action across similar tasks and observe the same outcome. For this reason, upon entering the label name, the user can press their ‘enter’ key to submit the label. This shortcut key is consistent throughout the application, applying also to editing a label and also when a user is selecting an image to open.

**3. Flexibility**

Flexibility is mainly focused on evaluating how well the interface can adapt to different users and their specific use cases. This may be through examining how easily customisable the application is or how easy it is to fit the specific task flow of a given user.

**3.1 Dialogue Initiative**

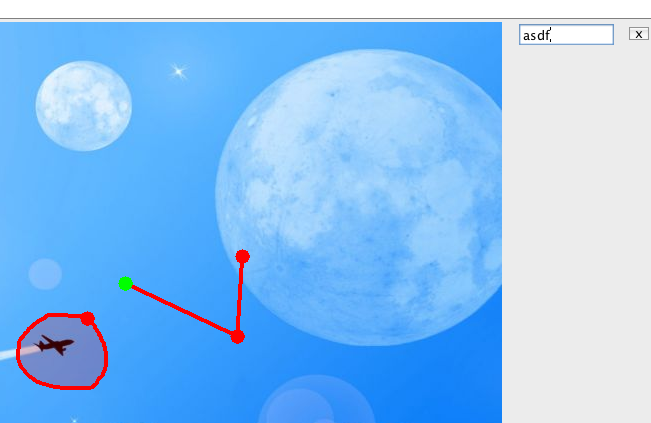
The dialogue initiative principle dictates that dialogues should be designed for closure. A good example would be the Chart Builder Wizard in Microsoft Excel which navigates the user toward the end of the dialogue by allowing them to click one button to navigate through all the steps.

To incorporate this principle into our design, we ensured that our two dialogues, the label prompt and the open image browser, are easy for a user to close.

In the label prompt, the user can type in the desired label and press enter to finish the label or press escape to discard the polygon and start from scratch. The dialogue is also very simple, containing only a text field and two buttons. This reduces the complexity of this choice, ensuring that the user will be able to choose his or her path quickly.

**3.2 Multi-threading**

It is also important to respect the preference for multi-tasking many users have. To allow users to only complete one task at a time is limiting their ability use the application effectively.

To solve this problem, we allow users to perform multiple tasks at one time and maintain state in the case that a user switches from drawing a polygon to editing a label name. While finishing the polygon, the label being edited stays active and the user can resume editing it once he or she has finished drawing the polygon.

**3.3 Task Migratability**

The task migratability principle evaluates whether repetitive tasks are performed automatically by the computer whenever possible.

To address this issue we have ensured that the user is not required to traverse through several menus to save their labels. Instead each time a user adds, edits or deletes a polygon or label, the application automatically saves the changes. Handing over the repetitive saving task to the system allows the user to focus on their job of creating labels.

This is slightly unexpected behavior (as users are more familiar with using save dialogs when dealing with documents), however save dialogs add another task for the user. Moreover, giving the user the option to save instead of auto-saving, opens the potential for a very frustrating situation where the user accidentally discards hours of work.

In more complicated applications where the potential for a user to want to discard his changes is moderately high, auto-save is more arguable, but here too the paradigm seems to be shifting.

In OS X 10.7 (Lion) Apple implemented AutoSave across the entire operating system. Noted apple reviewer John Sircusa wrote “For those of us who understand the pre-Lion document model and have been using it for decades, the idea that we are no longer in control of when changes to open documents are saved to disk seems insane.” However, he ultimately concludes that the Autosave model is better from an end-user perspective. For a dramatically simpler application like the Image Labeller, auto-save is therefore an obvious choice to improve simplicity.

Finally, the average user will likely rarely use the delete feature, instead focusing on creation (using undo during the occasional slip or mistake). Since the primary action is creation, saving will be more likely to create the need to delete unwanted labels instead of making desired ones irretrievable.

The benefit is another (irritating) step removed when labelling images.

**3.4 Substitutivity**

The substitutivity principle argues that that equivalent values of input and output should be allowed to be substituted for each other. (or allowing the user to complete one task in multiple ways)

To address this principle, we have ensured that with each and every menu item, there is an associated keyboard shortcut. This allows experienced users to use the shortcuts instead of clicking on the menus which can reduce the time taken to complete the task by a significant amount.

Moreover we allow the user to decide whether to draw by establishing points or through a freeform tool. Both output the same result on the image and permit the user to choose which method of input suits him or her best.

**4. Robustness**

Robustness evaluates the quality of the cues that the application gives the user to indicate that he or she is moving towards their goal. It also evaluates the ability of the user to recover from slips and mistakes.

**4.1 Observability**

The observability principle evaluates the user’s impression of the state of the system at a given time.

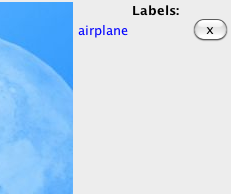
To address this principle, we have implemented several interface enhancements to show both the state of the system and the available actions within a certain state.

When the user clicks on the image to begin a new polygon, the first point is coloured green. This colour indicates the initial starting point and stays green until the polygon is completed. This tells the user that he or she is currently drawing a polygon, and clearly indicates the point to click to complete it. This is especially important since it helps to mitigate the clarity lost by no longer having a large “new object” button.

Additionally the first and final points of a polygon will turn white when the user hovers over them. The change of color clearly identifies that clicking on either of these points will produce a nonstandard action (something other than drawing). In the case of the start point, clicking will finish the polygon. Clicking on the the last point allows the user to freeform draw starting from that point.

Another example, mentioned before, are the availability of Undo and Redo. When the user is unable to use these commands their menu items are greyed out and the shortkeys disabled.

In order to make the editing of existing labels more usable, mousing over image labels highlights the corresponding polygon by filling in the enclosed shape in slightly transparent red. This enable the user to quickly identify which polygon they are currently changing the label for. The label text also changes color to identify that clicking on the text allows the user to edit the label name.

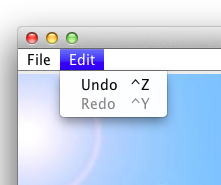


Finally, when opening a new image file, the user is restricted to selecting only files with valid image extensions, thus reducing the number of choices (and thus the time to task completion) when selecting an image.

**4.2 Recoverability**

The recoverability principle evaluates how well a system supports recovering from slips and mistakes. This means that the user should be able to take corrective action once an error has been recognized and revert back to any point in time (within reason).

To address this principle, we implemented “Undo”. The Undo functionality in our Image Labelling application allows a user to undo drawing, and creating or deleting of individual polygons and labels. The feature is accessible through both a menu item and a shortcut key. Undo ensures that a user can recover from both slips (which are common in a painting application) and mistakes, when the user is initially becoming familiarized with the interface.

Building on the recovery feature, we have also implemented Redo, which protects against slips when using undo. This further improves recoverability.

Both features ensure that a user is not quickly frustrated and forced to repeatedly redraw complete polygons whenever he or she makes a mistake. Furthermore, in addition to time saved from not having to redraw complete images, the combination of Redo and Undo enable the user to draw more quickly since the time cost of making a slip has been dramatically reduced so one can be less cautious when drawing.

**4.3 Responsiveness**

Responsiveness demands that actions that are similar in nature should take roughly the same amount of time to execute. For example, undoing and redoing an action should take roughly the same amount of time.

Since the application is simple it was relatively easy to maintain consistent response time throughout. It does however begin to slow down when too many labels are applied to an image. We therefore limit the number of labels on a given image to 15 to try to maintain the responsiveness of the interface.

**5. Future Extensions**

Due to time restrictions, we could only implement a finite number of improvements to the Image Labelling application’s interface. Some potential extensions include:

* Switching images from the same folder by pressing the left and right arrow keys
* Merging multiple overlapping polygons
* Dragging existing polygons
* Changing colour of polygons to help distinguish between them
* Zooming in and out of images
* Interaction with a server to store improvements remotely

**6. Conclusion**

Our Image Labelling application provides a simple and intuitive interface for the user. It makes the core purpose of the application, drawing and labeling objects, as quick as possible, reducing the entire process to three unique steps: drawing the polygon on the image, typing the text for the label, and then pressing “submit”.

The rest of the interface stays out of the way unless the user seeks it out, with the interface providing only a few tips at the beginning intended only for power users. It is therefore applicable to a widely diverse user-base.

By reducing interaction with menus, automating tasks, providing recoverability support and building upon the user’s pre-defined conceptual ideas, we have created an intuitive Image Labelling application that inherits from the design principles of UNIX applications and is based upon the notion of simplicity.