GR2 Designs

David Way, David Hayden, Joseph Lim, Carl Vondrick

Scenario

Sally is a college freshman. She notices that she spends a lot of time walking to and from her dorm and classes, and so she decides to purchase a bike. She does not know much about them, and is not interested in spending a lot of time looking up reviews at home. Instead, she simply goes to the nearest bike shop.

With assistance from a salesman, she tries a number of bikes, and eventually narrows down her options to several different brands and models. She likes all of them, but she cannot decide. She does not want to just ask the salesperson because she fears that he is biased. Although she is not necessarily a technological innovator, she is very proficient with modern technology. So, instead, she takes out her iPhone and opens Schnap It!

By opening Schnap It!, she aims to accomplish three goals: 1) get information about bikes, 2) compare bikes, and 3) be assured she is getting unbiased reviews. We now present three designs that address these points and give analysis for their advantages and disadvantages.

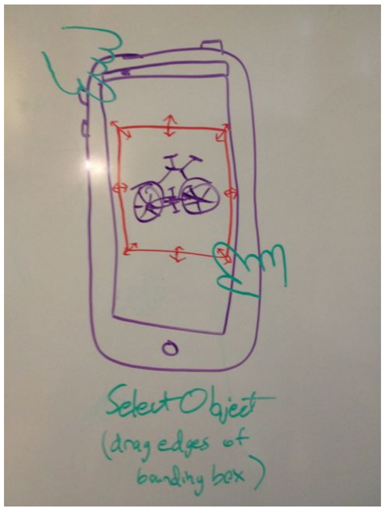
Design A

Sally individually takes a picture of each bike:

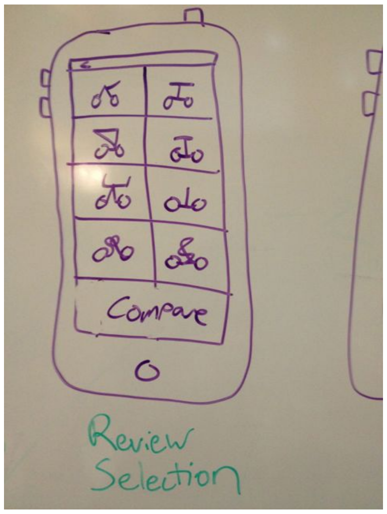
Shoot
Take Photo

...

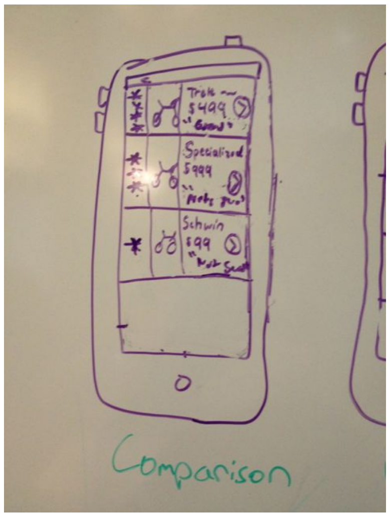
And then quickly identifies where in the picture the bike is by using gestures to draw a bounding rectangle:



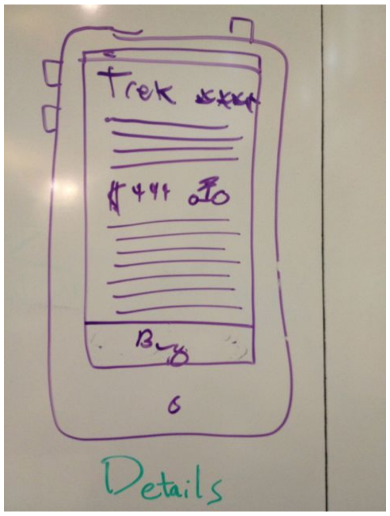
All her pictures are aggregated into a grid, where she can select multiple which to review:



Having selected the bikes of interest, the program opens a “reviews” shopping cart, which contains a row for each selected bike, along with its automatically-determined brand and model number, its lowest price from a reliable vendor, and its rating from a reliable reviews site. Sally notices that although the Schwin (bottom row) is only $99 online (it is $169 at the store), that it is also only rated one star.



She decides to get more information about the expensive, but highly-rated Trek bicycle. To do so, she simply taps on the arrow in the Reviews Shopping Cart. This brings her to a page that shows price, description, and collections of reviews. The same page also shows the sources of the reviews so that she knows who is influencing her buying decisions. After looking over it, she is decides to buy the Trek, there in the store.



Analysis for Design A

Efficiency

In this design, the user has a different screen for each decision. The user also must individually take pictures of each object. Then when all of the pictures have been taken, the user decides which objects to review together. This design is inefficient design since the user is not likely to take pictures of products she is not interested in. Efficiency might be improved by aggregating the “snap picture” and “choose objects to review” sub-tasks into less screens. For instance, one picture could be taken and the very next screen shows the shopping cart with information about each object specified.

For the purposes of this application, the user must spend time annotating the picture to specify which object to review. The method of annotation in this design is very efficient. The method of comparing objects is moderately efficient; if the user wants to compare a specific detail about two objects they must travel back and forth between detail screens. The user does not have the ability to move that detail to the Reviews Shopping Cart where all objects and main details (such as stars) can be viewed.

Learnability

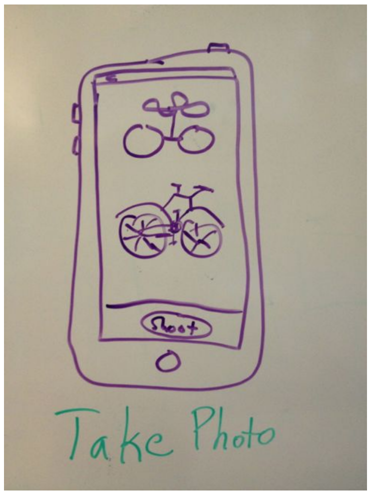
Given this application starts by going right to the camera, the user may not know exactly what to do the first time they use the application. We could use a first-use tutorial, or else let them learn by taking a random picture and going through the process to figure out how the application works. This latter scenario detracts from the initial learnability of the application because there are no clear instructions for the user to read. In the long run, the user does not have to bypass a repetitive start or instruction screen. After taking a picture, the user can touch randomly to learn exactly how to specify which object they are interested in. After specifying, the application reinforces exactly how to take pictures of object and annotate by allowing the simple process to be repeated. This also gives the user practice. Selecting which objects to review is fairly easy to learn. The pictures selected on the grid will be highlighted to provide a selection affordance to the user. The arrows on the Reviews Shopping Cart clearly afford a page of additional details. In all, the learnability of this design is fairly good. It allows the user to learn by experimentation and stays out of their way.

Safety

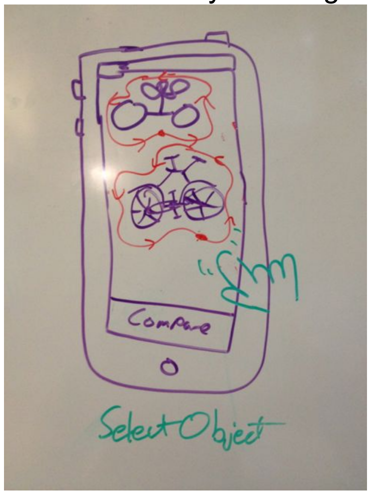
Since the efficiency of iPhone applications is very good, the user can correct mistaken view changes fairly quickly by hitting the back button to move up the screen hierarchy. Software architecture on the iPhone make this easy to implement and allows the user to efficiently recover from errors. The user cannot modify information displayed about the objects, separating the backend model from the user model. To the user, the application allows a way to quickly look up products and a simple way of comparing specifics. Errors may occur in the backend, such as the object being unrecognized, but this design assumes that will not happen. Modifications could be made for the case when a new object is specified that is not in the database or when the picture taken is not of high enough quality to work for the object detector.

Design B

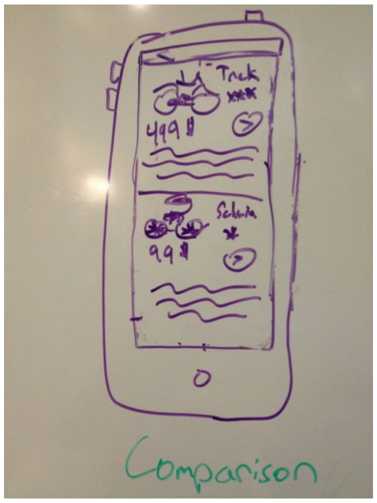
This time, Sally can collect information about all the products she is interested in more quickly by taking pictures that each have multiple bicycles in them:



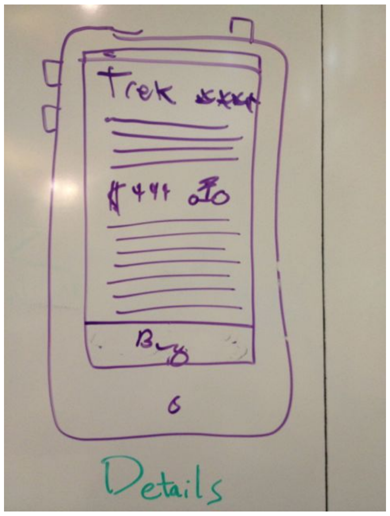
For each picture, she indicates where each bike by outlining them with her finger:



She is automatically taken to the scrollable Reviews Shopping Cart, where the bicycles of interest are aggregated, and show best, reliable online prices and reviews.



As before, she can look up more detailed information (description, multiple reviews and prices) about each bike by clicking on the arrow.



Analysis for Design B

Efficiency

The efficiency has greatly improved from design A. The user may not have to spend time taking *n* pictures for *n* products. There is no pre-review selection page since objects are automatically appended to the Reviews Shopping Cart. However, the tracing may take longer than the touch and drag method of Method A. There is no obvious way to modify the outline of the object - the user would have to restart tracing all over again. Apart from the more complicated annotation method, the efficiency of the application has improved because the user must make fewer decisions on less screens.

The efficiency issue of comparing a rare or usually unimportant detail still remains: the user would have to travel back and forth between Product Detail Views. Additionally, while allowing the user to annotate multiple objects in each picture could improve efficiency, it may not always be practical (e.g. you may not be able to arrange all products side-by-side). In this case, the user could simply take multiple pictures--the objects from each image are appended to the Reviews Shopping Cart View.

Learnability

Learnability has decreased in some respects compared to design A. The user may not know that they can specify multiple objects in each frame.They could learn by accident, or by watching someone else do it. Regardless, we could improve on this by having an object counter in the annotation view. We could also add certain affordances, like animations drawing sample figures around objects. Otherwise, the learnability is similar to or better than Design A because they do not have to select multiple objects from a grid.

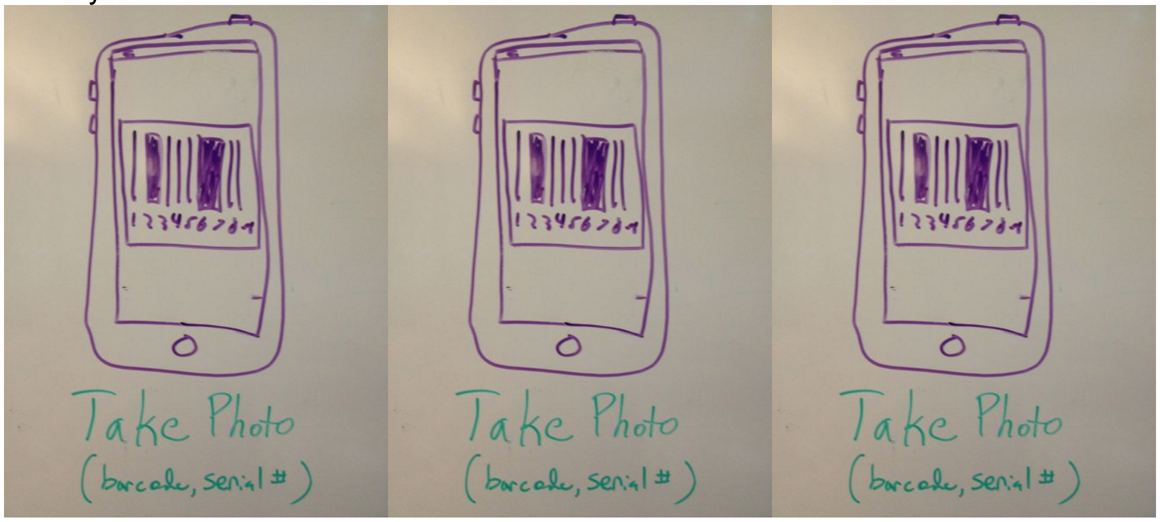
Safety

An large safety issue in this design is that the user cannot easily fix the drawn outline of the product. To fix a poorly drawn outline, the user must first erase the past outline or outlines because it visibility of the object. The user must then try again, with same probability of making another mistake.

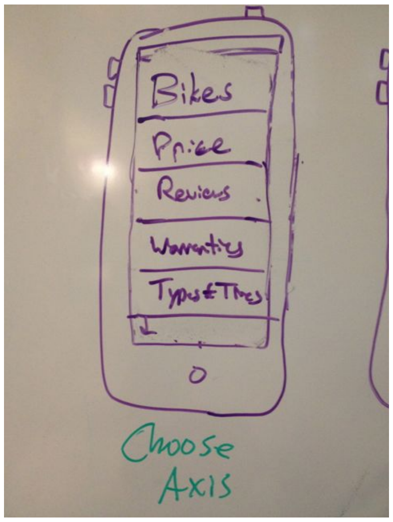
Often product may encompass or overlap each other in the image. In this use case, errors may occur in drawing and object detection. The user may not know exactly what to draw to specify such overlapping objects. For instance, the user may not know if one outline can potentially specify two objects? Perhaps part of the object is outside of the screen. The user does not know to cut off the object with the outline or if they can indicate that part of the object is not showing.

Design C

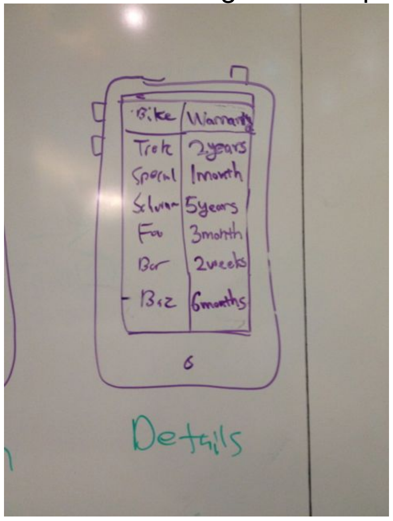
Rather than take a picture of the bike, she simply takes a picture of its barcode. This does not require any annotation.



Once Sally has taken pictures of each barcode, she can compare them in a table by price, reviews, or warranty. Since the software can determine that bikes are being compared (based on the product classification of each bar code), it can also provide additional values that can be compared on, such as construction materials (e.g. tire type). Sally is most interested in comparing the warranties for each bicycle, so she taps that:



...And is taken to a table where each bike she imaged is compared.



Bike	Warranty
Trek	2years
Special	1month
Schwinn	5years
Foo	3month
Bar	2weeks
Baz	6months

Analysis for Design C

Efficiency
This design is very different from the previous designs because now users only take pictures of barcodes. This is advantageous because it may be difficult to take a photo of the object due to occlusion phenomena. However, the trade-off here is that it may be difficult to find barcodes on some objects (e.g. cars). But, the added power of barcodes is that it gives us more precise lookup ability. In this way, the barcode can pull information directly from the manufacturer rather than relying on a third party look up service---this property significantly eases user's concerns that reviews may be biased. This storyboard also shows a third new comparison screen where the users get to pick which axis they want to compare on and see information about that. The new comparison interface allows the user to compare specific fields without traveling back and forth between Product Detail Views. Moreover, the look up service significantly improves

efficiency, but only if there is a barcode handy. If there is no barcode, the applicaton would have to default to some other type of annotation.

Learnability
The learnability of this design is high. Instructions to take a picture of a barcode would be clear. From then on, the results of the application are presented in a clear manner. Neatly laid out buttons describing different axis tell the user how results are organized. Bevels on these buttons allow for affordance, which indicate to the user that touching the arrows on the buttons will lead to an explanation concerning that detail.

Safety
The major fallback of this design is its dependability on barcodes for recognition. Several problems can arise from examining barcodes: the barcode may be scratched, in a unrecognizable format, or plainly absent. Products often have several barcodes on the packaging, such as one for the manufacturer and one for the seller. Users have no way of telling which barcode is correct. If the user snaps a photo of the wrong barcode, they will receive possibly incorrect information without warning.

Conclusion

We have presented three radically different user interfaces, their storyboards, and analysis describing their advantages and disadvantages. All three interfaces would allow Sally to find product information, compare products, and find unbiased information since each has high learnability, efficiency, and safety.

However, we hypothesize that the second interface (Design B) will be preferred by most users since its efficiency is the strongest. Several other features from Design A or Design B could also prove useful. Throughout this project, we will conduct further experiments to verify this hypothesis.