a conceptual
3d modeler for
the iphone

SKETCHPAPER

MULTITOUCH ELECTIVE 2009 | PROF. ROLF KRUSE

SEBASTIAN GERHARD	713 036
DENNIS PRASCHAK	713 051
DOMINIC SZABLEWSKI	713 057

INDEX

28.09.2009

CREATIVE RESEARCH

Our first step was the decision for a development platform. We opted for the iPhone or respectively the iPod touch. Not only because its nowadays kind of ubiquitous, but also because it is challenging to transfer traditional pieces of software to this, compared to desktop computers or dedicated multitouch tables, tiny device.

the iPhone in its current iteration

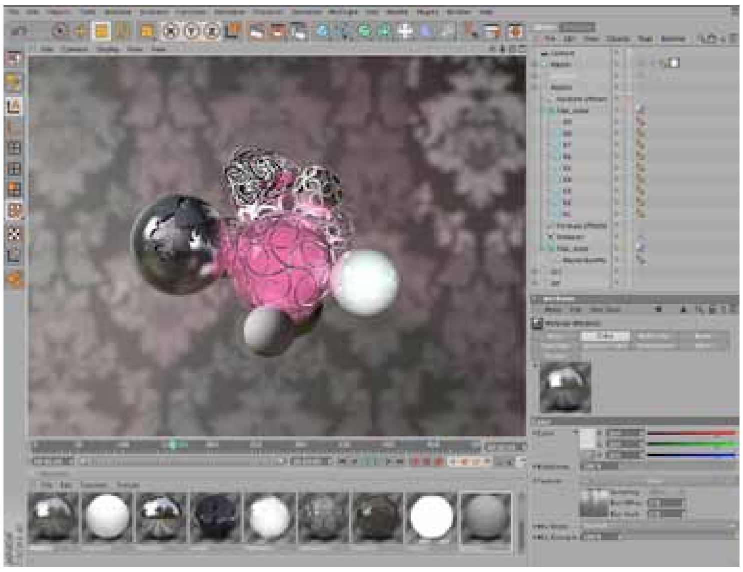
After lining out a couple of ideas, from origami to an advanced remote control for entertainment devices, we decided to develop a 3D modeling application.

Multitouch devices allow the user to manipulate content directly on the screen, using hands and fingers.

„Modeling“ means shaping and creating objects and structures. Think about a potter who creates pots and mugs by folding and modeling clay on his wheel.

3D modelers for desktop computers are complex software products, stuffed with complex uis and tons of viewing modes and visual aids. They are not like word processors which you can „learn“ in a short amount of time.

Most of them don‘t implement metaphors and their respective mental models. 3D modeling computers feature


MAXON Cinema 4D

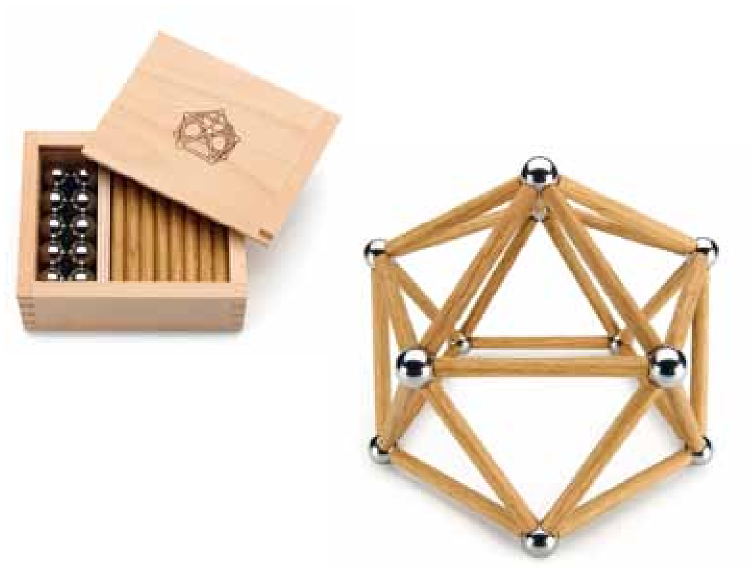
large screens and precise input devices. So it sounds like a contradiction in terms to develop such a piece of software for a tiny device with a 2,5“ screen and your thick fingers as the major input device.

But knowing how to handle the software is only half the battle. In order to create stunning three-dimensional objects, you need a handful of visual thinking abilities.

Everyone may experience that when trying to sketch even simple three-dimensional shapes on a slip of paper.

So we began research, looking for a design concept that combines playfulness and a learning curve for your visual thinking abilities.

There are a lot of „toys“ out there offering exercises for you right hemisphere and your creativity by construct objects in finite space.



classic 3D construction game

Even simple building blocks for children provide you with a grid implied by the sizes of the individual blocks.

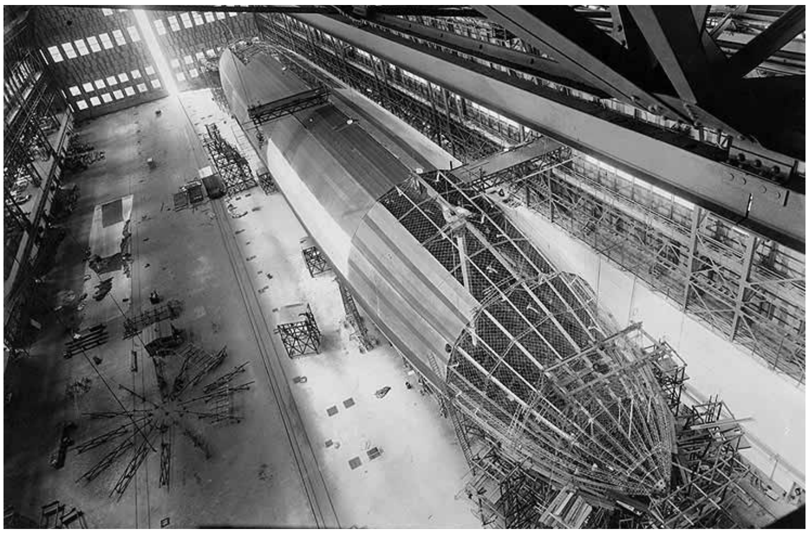
We found a collection of sophisticated 3D construction toys and grids. Inspired by the construction principle of airships, we made up our construction framework.

CONCEPT & INTERACTION IDEAS

At all conjunction points in the three-dimensional grid, touchable markers are displayed. If the user touches three of these distinct markers in a row, a triangular plain is created between those 3 points.

With this simple and easy to learn input “gesture”, the user should be able to create complex objects and forms.

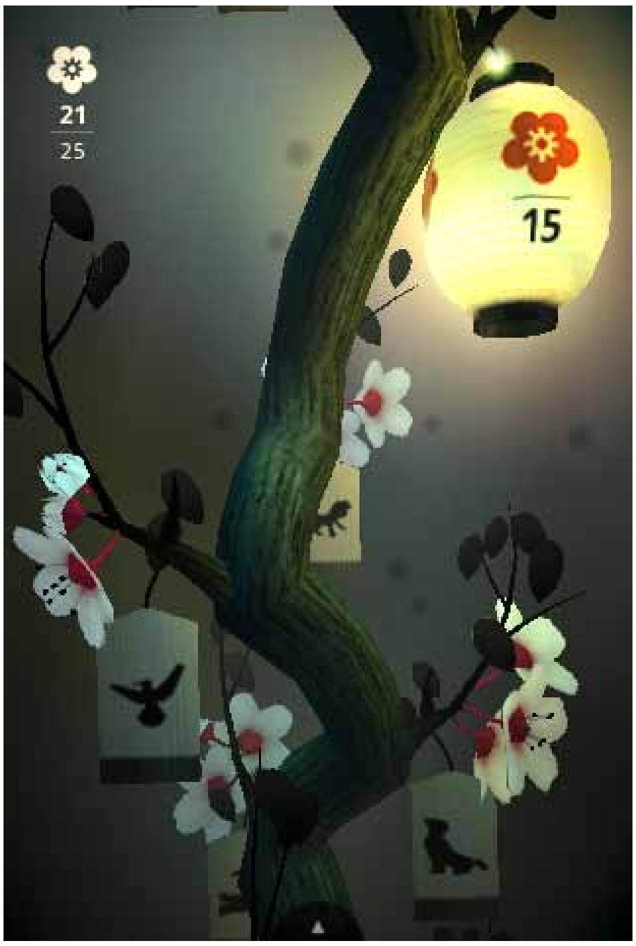
Selecting and editing objects could work the same way and were subject of the prototype testing.


airship construction in lakehurst/USA

The grid can be rotated with a two finger swipe. The zoom level can be adjusted using the iphonish pinch gesture.

Regarding the visual design, we had several ideas that made it in a testing session. Again, we played with the idea of the skeleton design of airships.

Another visual inspiration was drawn from a popular artistic iPhone game called "*zenbound*".



zenbounds' paper lanterns

"Zenbound's" paper lanterns are illuminated from the inside and generate a visual mood in contrast to the dark background.

We finally decided to make use of the iPhone's integrated camera. Enabling the user to use the camera to snap a photograph, or let him use pictures taken earlier as textures for the models, makes the whole modelling process more appealing and encourages to remix real-world objects with the application.

photograph of wooden shingles

USER SCENARIOS

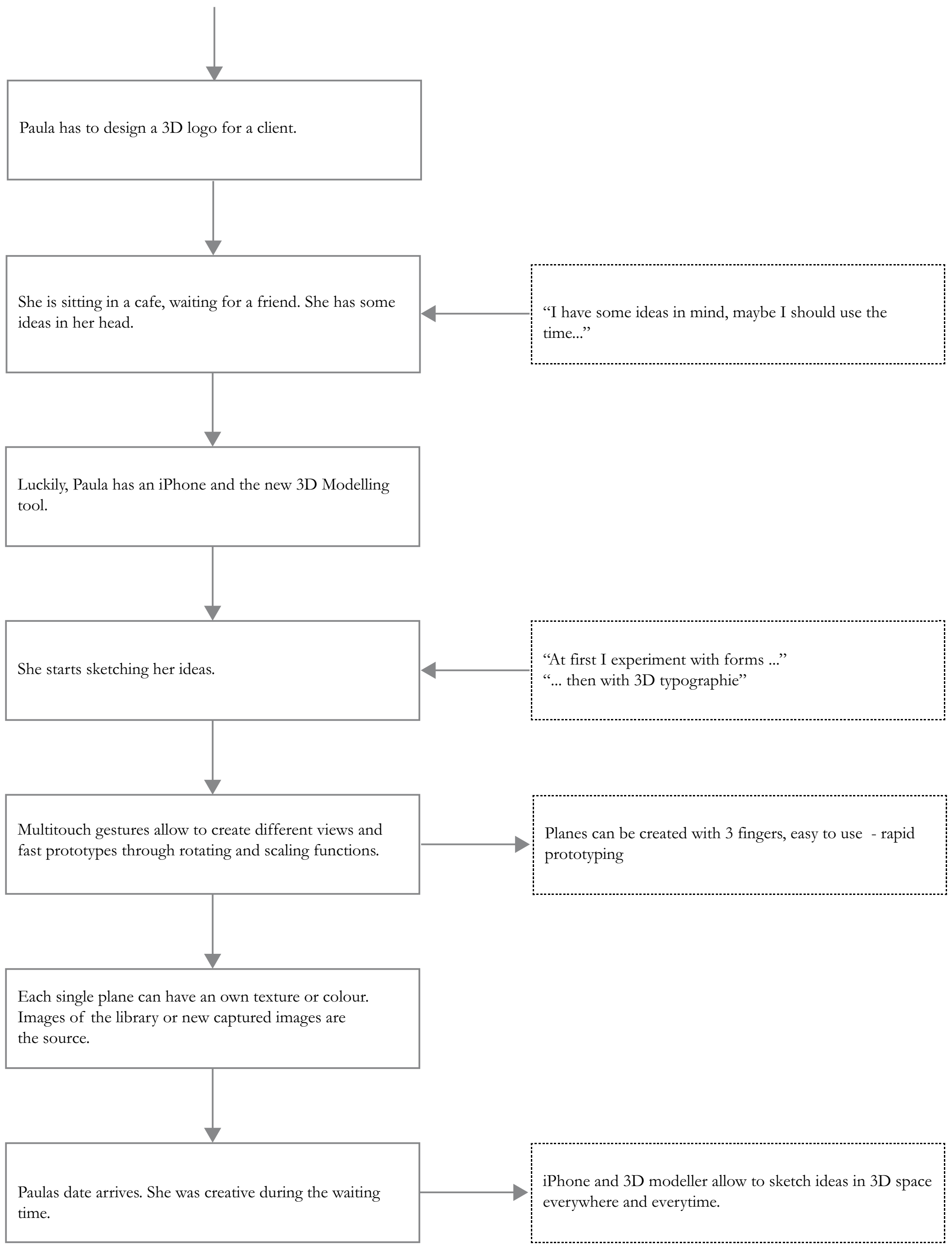
Writing user scenarios is a common tool in user experience design to collect information about motivations, actions and goals of users.

It helps to describe when a product is used for delving into user behaviour and understanding their real needs. Complete new features or improvements in interaction design can be found. User scenarios are also a very good help to communicate the functionality of a product both to customers and internal between different departments.

See the next pages for the user scenarios.

USER SCENARIO 1

Multitouch Elective: 3D Modeller User scenario 1



Paula has to design a 3D logo for a client.

She is sitting in a cafe, waiting for a friend. She has some ideas in her head.

"I have some ideas in mind, maybe I should use the time..."

Luckily, Paula has an iPhone and the new 3D Modelling tool.

She starts sketching her ideas.

"At first I experiment with forms ..."
"... then with 3D typographie"

Multitouch gestures allow to create different views and fast prototypes through rotating and scaling functions.

Planes can be created with 3 fingers, easy to use - rapid prototyping

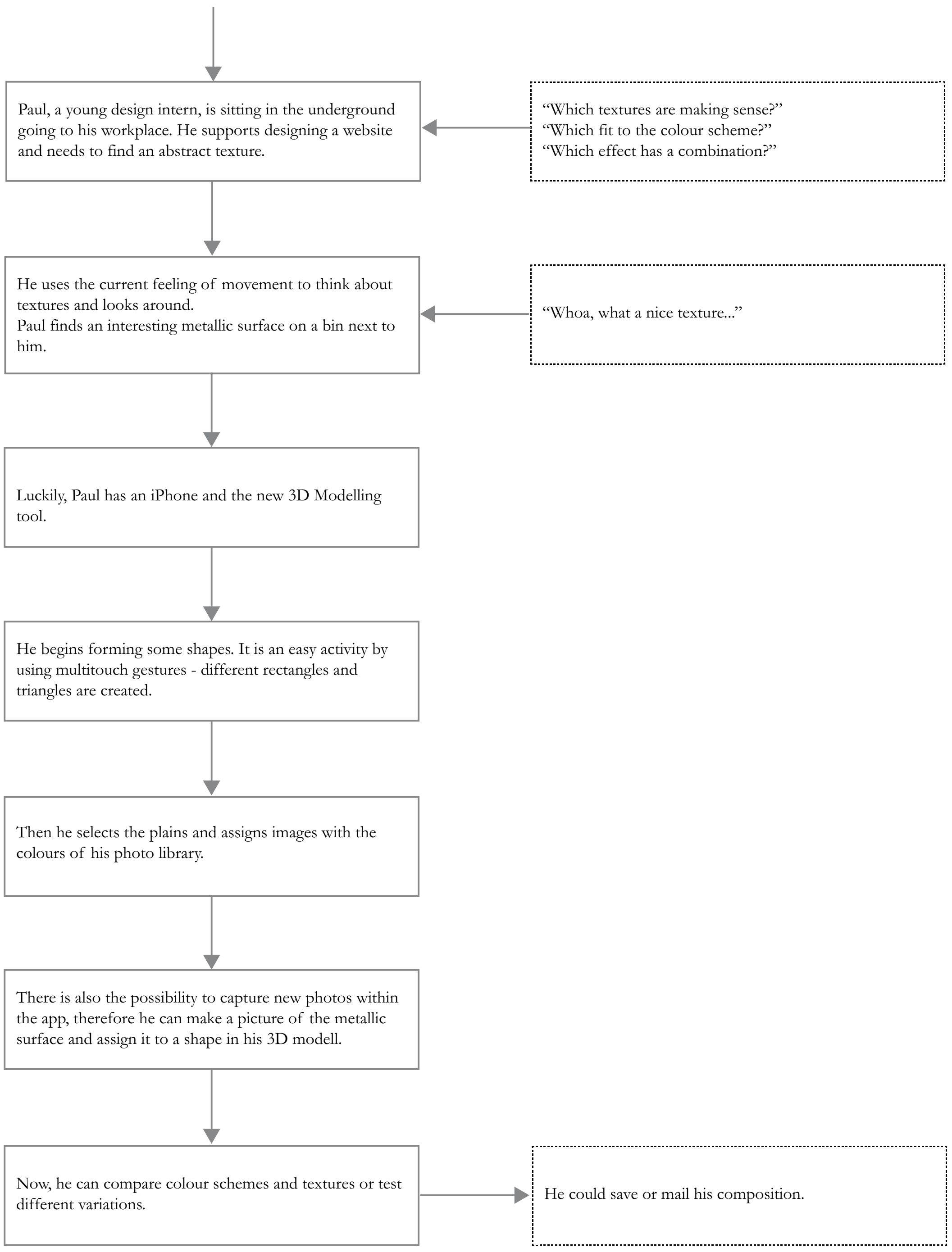
Each single plane can have an own texture or colour. Images of the library or new captured images are the source.

Paulas date arrives. She was creative during the waiting time.

iPhone and 3D modeller allow to sketch ideas in 3D space everywhere and everytime.

USER SCENARIO 2

Multitouch Elective: 3D Modeller User scenario 2



Paul, a young design intern, is sitting in the underground going to his workplace. He supports designing a website and needs to find an abstract texture.

"Which textures are making sense?"
"Which fit to the colour scheme?"
"Which effect has a combination?"

He uses the current feeling of movement to think about textures and looks around.
Paul finds an interesting metallic surface on a bin next to him.

"Whoa, what a nice texture..."

Luckily, Paul has an iPhone and the new 3D Modelling tool.

He begins forming some shapes. It is an easy activity by using multitouch gestures - different rectangles and triangles are created.

Then he selects the plains and assigns images with the colours of his photo library.

There is also the possibility to capture new photos within the app, therefore he can make a picture of the metallic surface and assign it to a shape in his 3D modell.

Now, he can compare colour schemes and textures or test different variations.

He could save or mail his composition.

INTERACTION CONCEPT

The whole construction process is based on selecting and de-selecting the marker points. We agreed upon the follwing set of combined gestures:

switching the view

one-finger-swiping rotates the grid

two-finger swiping moves the grid according to its current orientation

creating a plane

single-tapping a first marker point selects and highlights the first point.

single-tapping a second marker point selects and highlights the second point.

single-tapping an already selected marker point again toggles its selection state.

single-tapping a third marker point finally spans a plane between all three previously selected marker points.

shaking the iPhone deselects all previously selected marker points.

selecting a plane

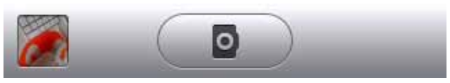
*double-tappin*g a plane lets you select a plane. The plane gets then automatically centred in the viewport.

editing a plane/adding textures

after selecting a plane, the texture image can be edited. *Pinching & spreading* two fingers on the texture image lets you zoom in and out. Rotating the two fingers rotates the image.

adding textures to a plane

When a plane is selected, the iPhone camera bar fades in an gives you access to your film roll and the camera. After selecting an existing photograph or snapping a new one, the plane is updated with the new texture.

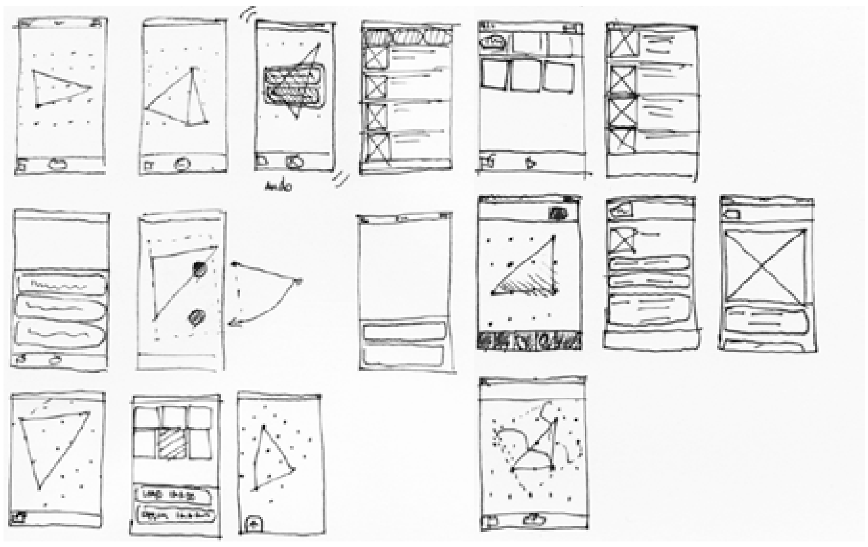

iPhone camera bar

deleting a plane

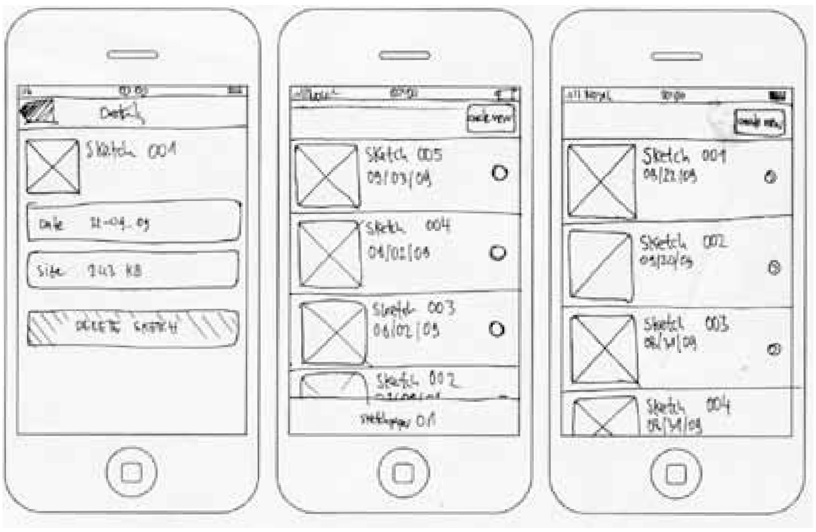
Shaking the iPhone when a plane is selected deletes the selected plane.
(Shake-to-undo is a gesture introduced to the iPhone in the latest software update.)

UI DESIGN & TESTING

We opted for the iPhone as our development platform. The iPhone UI is designed an structured by the Apple Human Interface Guidelines which provides clear rules and hints for ui elements. We started to line out the interface using sketches. These sketches helped us to line out basic interactions between the application and the user.


first ui sketches

We created some use cases for the application and tested them against our sketches. We discovered some usability glitches and stumbled about a bunch of enhancements that made it in our second, more sophisticated series of ui sketches. In these second sketches we focussed more on the graphical aspects of the interface.



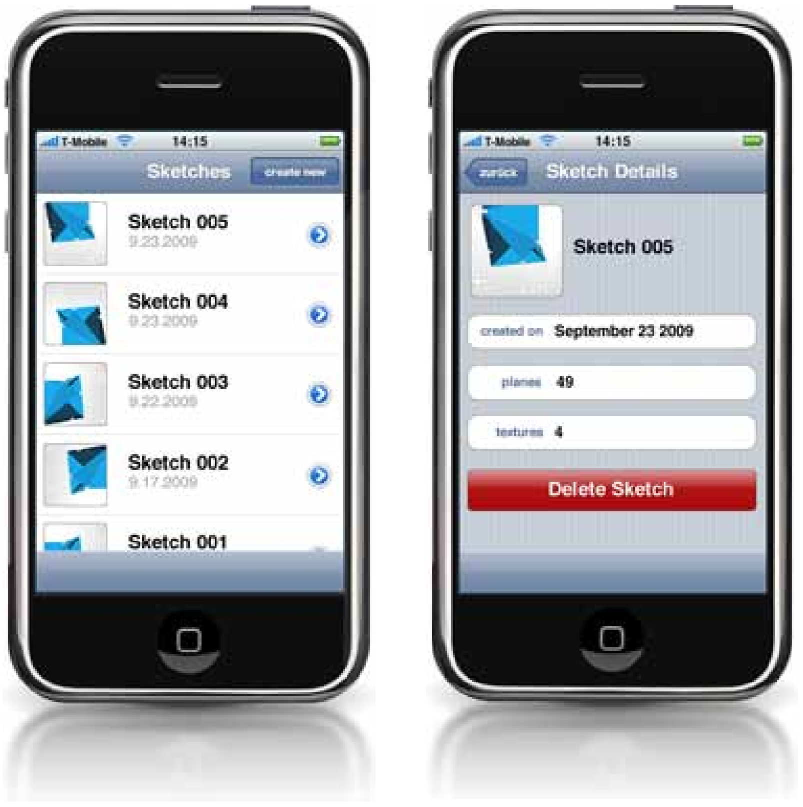
excerpt from final ui sketches

Using the graphical elements of the iPhone os, we translated the features of our application in a real gui.

Once more, using pen and paper helped us to concentrate on the character of our application. Testing and "implementing ui elements without the final aesthetics in mind, helped us to streamline the whole development process.

When we switched to photoshop and apple's interface builder, we already knew what do next. We already knew which ui elements to use in which situation.

We designed the menu and all dialogues.
See the next page for some screenshots.

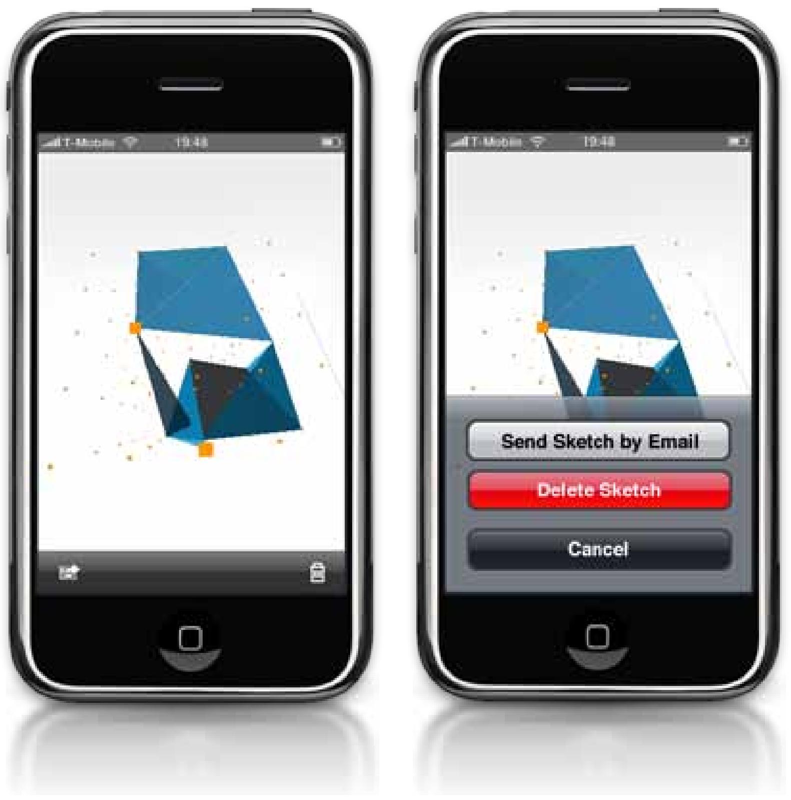


final ui design/ menu

The screens above show the home screen of the application and the sketch detail view. The “other half” of the ui design is the editor or modeling view. During the research and concept phase, we decided on the grid approach, featuring the skybox, the clickable conjunction points and the faces.

So we had to design these three elements alongside with some additional ui elements for the plane editor.

We gave the skybox some orientation by adding a slight gradient and a noticably darker bottom plane to it. The conjunction points exist in two states: *“selected”* and *“standard”.* These two need to be easy to distinguish, also when a lot of conjunction points lie between two selected points.



*final ui design/
modeler view*

Enlarging the selected points sever the purpose very well. The minimal skybox design aligned very well with the custom textures future users will be able to add to the models by using the iPhones built-in camera.

Saving and deleting a sketch can easily accessed by the context sensitive menu, which pops up when the lower edge of the screen is touched for a second.

TECHNICAL IMPLEMENTATION

Creating a 3D modeler for the iPhone is challenging for several reasons. The device provides no hardware buttons to directly interact with the application and because of the small touch screen there is no margin for input errors.

All inputs have to be carefully evaluated and checked for which action the user wants to perform. Another problem is the relatively slow hardware, compared to modern desktop PCs. Texture sizes and polygon counts can have a large impact on the applications performance.

Applications for the iPhone are programmed in Objective-C, a superset of the still omnipresent C programming language. It adds object oriented programming functionality and several other features, but is still every bit as close to the hardware as its predecessor.

Oftentimes this makes programming in Objective-C quite tedious, since many of the amenities of higher level programming languages are missing. However, it also makes Objective-C very fast, which is nice for such a graphically intense application.

Implementing a 3D view on the iPhone is not much different from one on a desktop PC. The iPhone natively supports OpenGL-ES, which is a subset of the popular OpenGL 3D-Graphics API. In OpenGL-ES many esoteric functions have been cut and most of the advanced features like pixel or vertex shaders are missing too. This however is not a big deal, since our application only displays very simple geometries anyway. Vertex arrays are used wherever possible to minimize the amount of draw calls.

Gesture Recognition

Though all the gestures incorporated in our application are very simple, distinguishing them from each other is difficult on such a small screen. In particular, distinguishing the pinch gesture from the two-finger-scroll was a challenge. Both gesture require the user to put two fingers on the screen and both gestures can start with these two fingers in any distance to each other.

We implemented a fairly large threshold to check whether the two fingers are moving in the same direction (scroll), or away from each other (pinch). Only after this is determined, an action is taken by the application. The application remains in that mode (scroll or pinch) until both fingers are taken off the touchscreen again. This turned out to work fairly nice and produces almost no unintentional results.

A normal single finger tap action is only then performed, when the finger is lifted again and didn't move while it was on the touchscreen. This allows us to also use one finger movement as input, while still retaining the tap functionality.

We used the iPhone's built-in accelerometer for the delete gesture, where the user has to shake the device to delete a face. Recognizing a deliberate shake was pretty straightforward. The accelerometer was set to a fairly low update rate of 10Hz and just checked for a large movement in either direction.

Tracing/Picking

When the user taps on the screen, he either wants to select a grid point or a face he already built. Thus, we have to determine which object the user wants to select, based on its current projected screen coordinates. There are several ways to do this. One of the easier (though quite performance intensive) ones, provided directly by OpenGL was sadly not available in OpenGL-ES.

We ended up implementing a picking function that creates a ray originating from the precise point the user tapped on the near plane and ending on the far plan. In order to make this work, the current rotation, zoom and position of the virtual camera has to be considered. The ray is then checked for intersection with each of the grid points as well as with all faces in the scene.

All the grid points are evaluated with an algorithm checking for ray/sphere intersection, while the faces are checked with an ray/polygon intersection algorithm. This gives us the distance to the closest face, as well as the distance to the closest grid point this ray intersects and ultimately the desired object to select.

PREVIEW

During our design and implementation processes we found a couple of useful features, which are not yet implemented in the current prototype. An essential feature is the possibility to save created models. This could be realised local, on the iPhone, or remote in an online community with shared webspace. An organised online platform could support an exchange of models between users or as a gallery. This would be a marketing instrument for the modeling application itself, too.

An export function for common mac and pc modeling applications would be a big benefit for users. Rapidly created sketches on the phone could then be edited and used on desktop machines.

Perhaps it would be a useful feature to add a curve option, which allows to create round forms - this step need to be well planned to secure the integrity of the whole application to not get too complex in use.