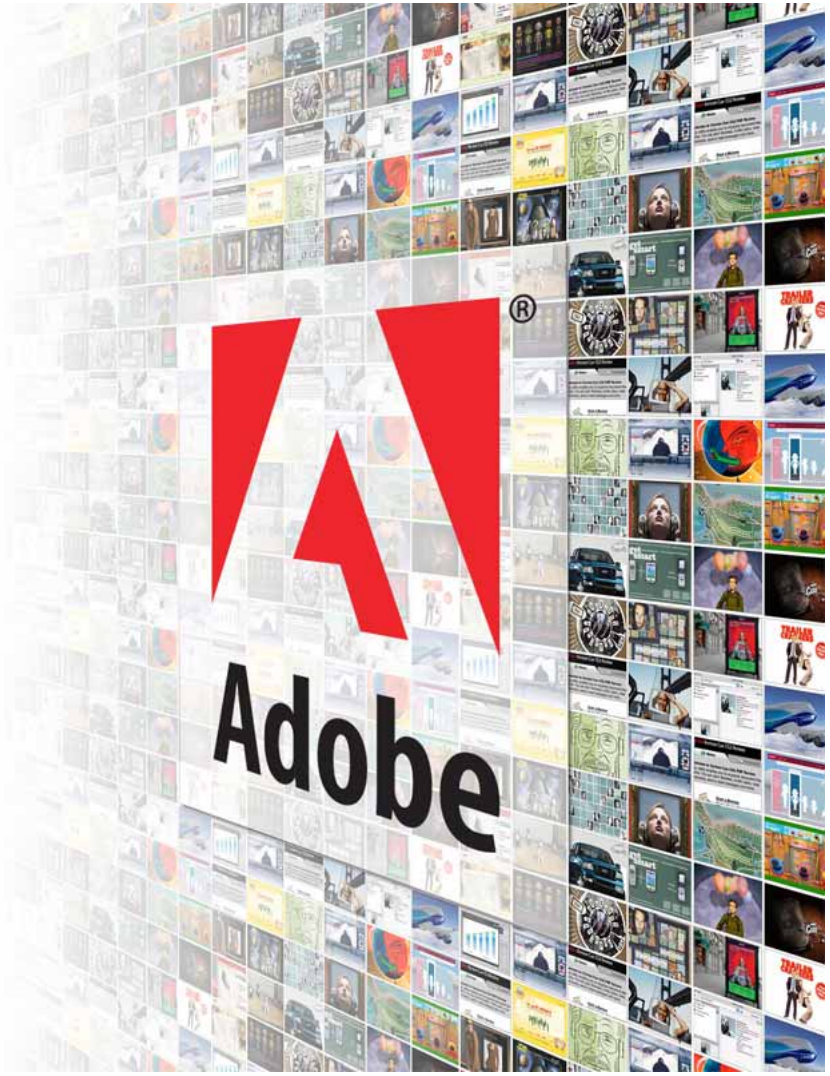
Science of 'Shrink Wrap'

A Look Inside Adobe® Photoshop®



Sean Parent

Principal Scientist & Manager
Adobe® Software Technology Lab

January 18th, 2008

Adobe

Adobe Today

Worldwide Offices

Corporate Headquarters – San Jose, California

Key Statistics	
FY2007	$3.16B
Years in Business	25
Employees	6,000+

Adobe

Photoshop History

- 1987: Started by Thomas Knoll
- 1990: 1.0 Shipped by Adobe
- 1991: 2.0 Clipping Path
- 1993: 2.5 First Version on Windows
- 1994: 3.0 Layers
- 1996: 4.0 Actions & Adjustment Layers
- 1998: 5.0 History & Color Management
- 1999: 5.5 Web Development
- 2000: 6.0 Typography
- 2002: 7.0 Camera RAW, Healing Brush, Natural Painting
- 2003: CS Lens Blur, Color Match, Shadow/Highlight
- 2005: CS2 High Dynamic Range Imaging, Smart Objects, Lens Correction
- 2007: CS3 Smart Filters, Improved Compositing Tools

Adobe

Demo

Photoshop Code

- **100% C++ since Photoshop 2.5**
- **Statistics for Photoshop CS3 (Core):**
  - **Lines:** **1,467,150**
  - **New Lines:** **107,129**
  - **Engineers:** **30**
  - **Develop Cycle:** **24 months**
- **Image Processing Code: ≈15%**

Q: Where is the other 85%?

Adobe

Q: Where is the other 85%?

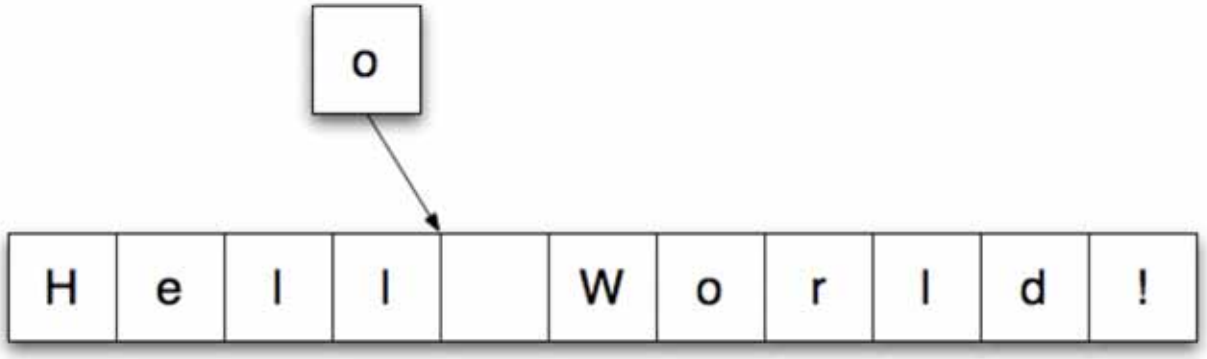
- **A: The User Interface**

The User Interface

- **Definition: A User Interface (UI) is a system for assisting a user in selecting a function and providing a valid set of parameters to the function.**
- **Definition: A Graphical User Interface (GUI) is a visual and interactive UI.**

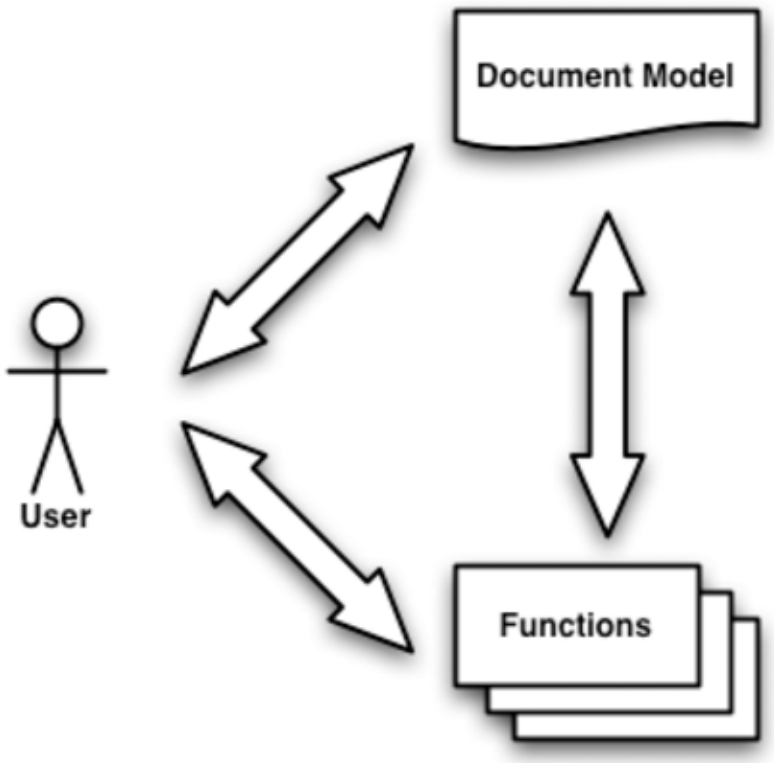
Example - Text Editor

- **Document Model**
  - **sequence of characters**
- **Functions**
  - **insert (sequence, location, character)**
    - **Precondition: location must be within the sequence.**
  - **erase (sequence, range)**
    - **Precondition: range must be within sequence.**



Adobe

Design Space



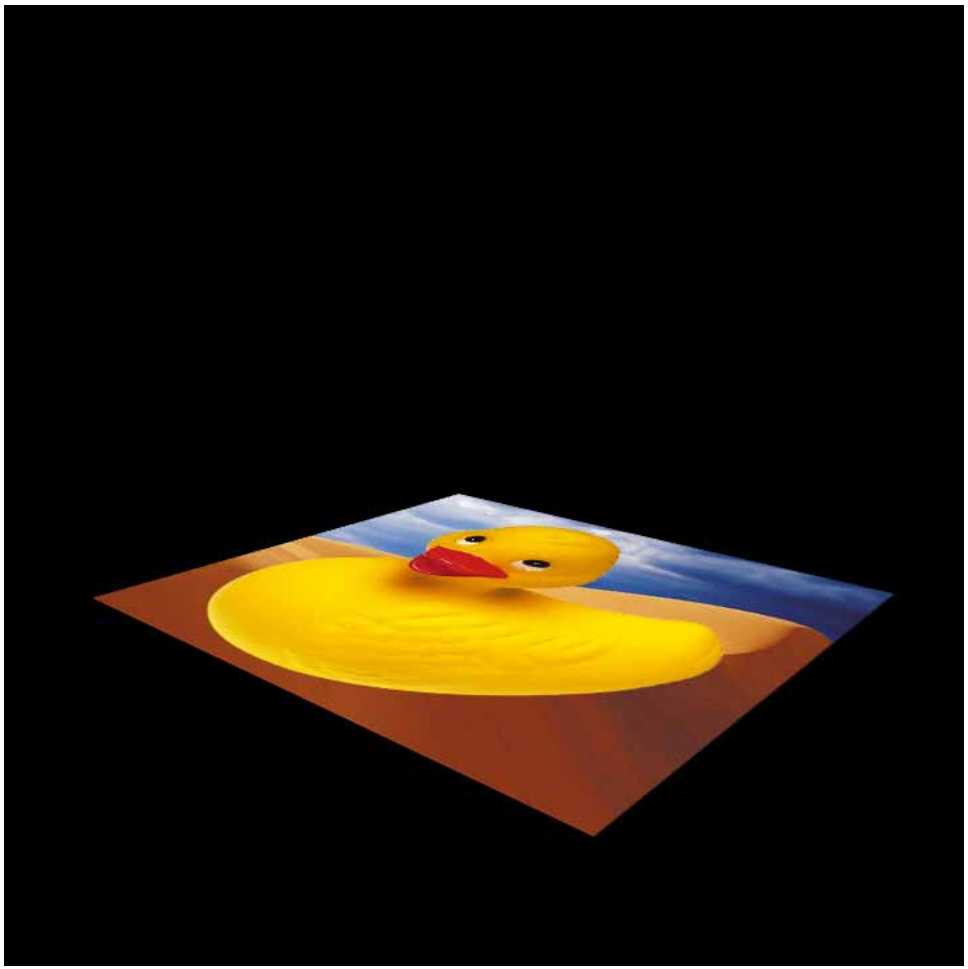
Adobe

Design Space

- **User requirements determine desired functions and model**
- ***Assistance* drives choice of algorithms for functions**
- **Choice of algorithms constrains choice of data structures**
- **Data structures and algorithms constrain available functions**

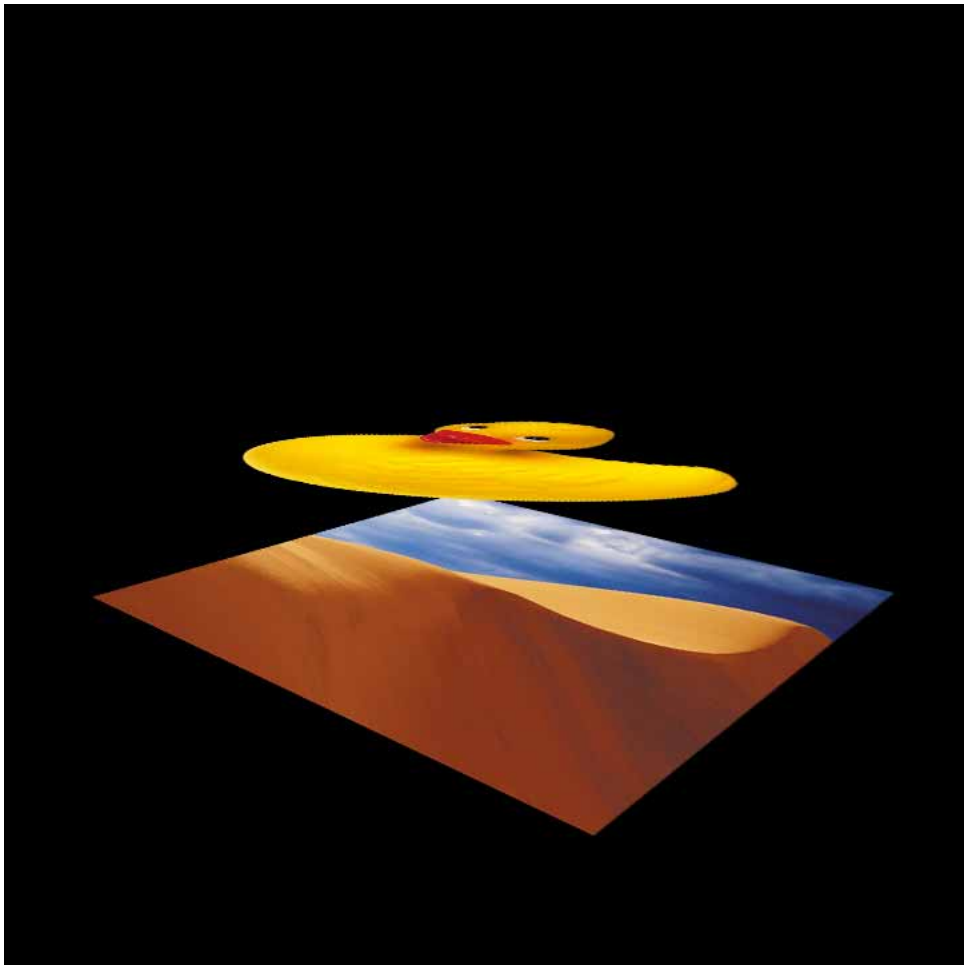
Photoshop Document Model

Photoshop Document Model



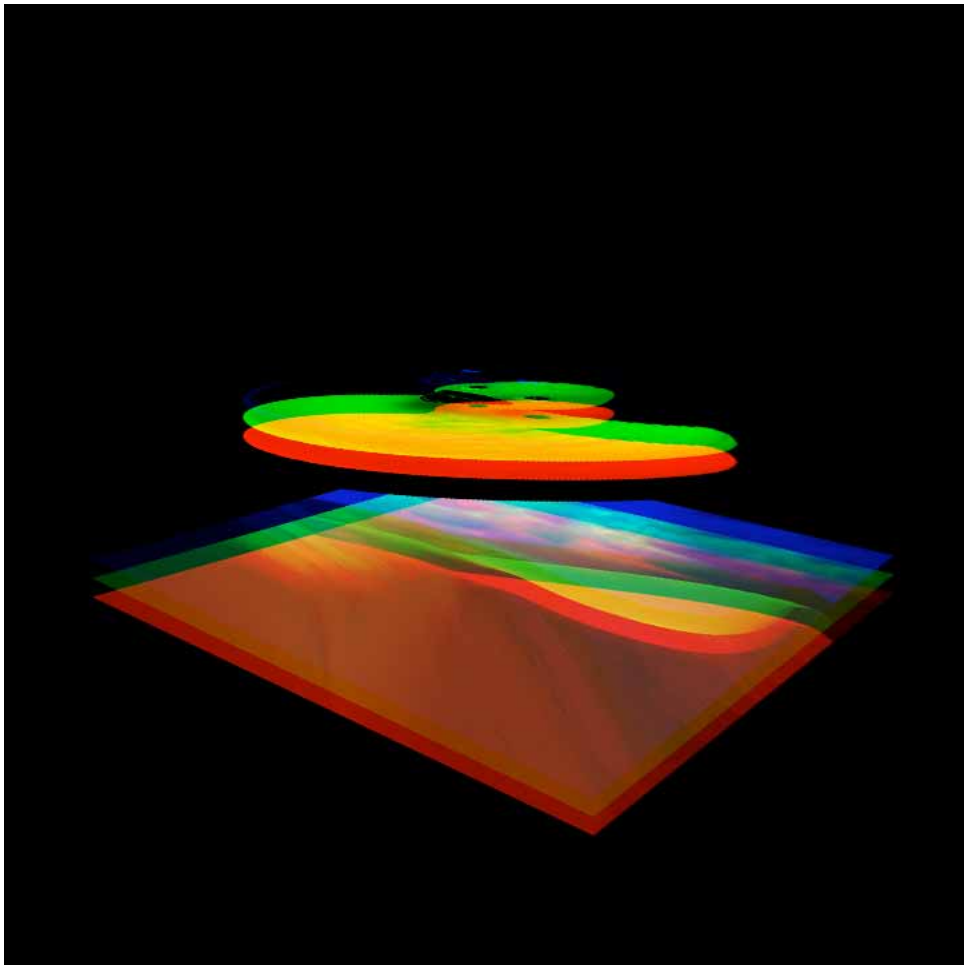
Adobe

Photoshop Document Model



Adobe

Photoshop Document Model



Adobe

Photoshop Functions

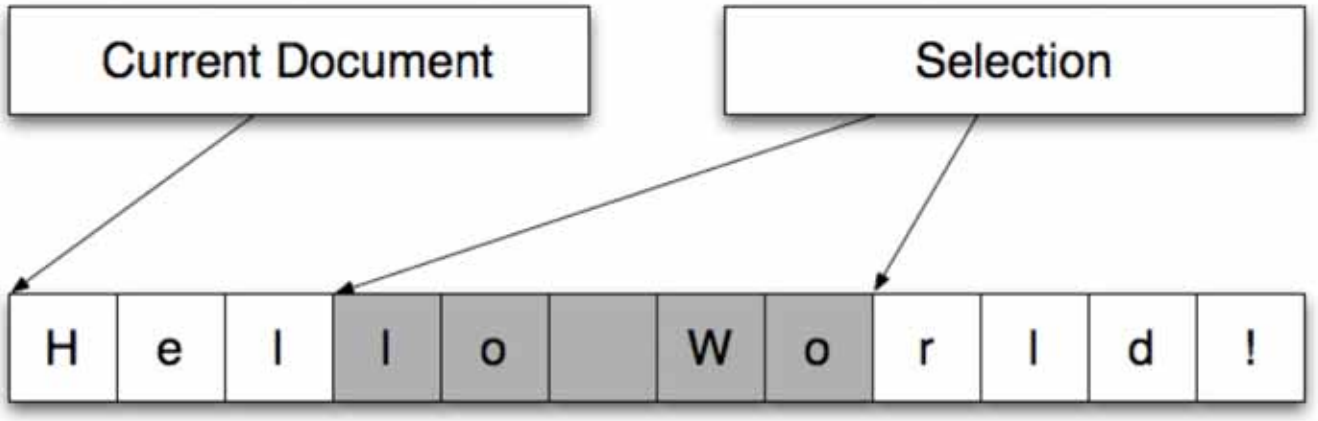
- **make_layer(), gaussian_blur(), transform_image()… and several hundred more.**

Providing Context

- **Often there is a "subject" which is being operated upon. Provide state to remember the current subject.**
- **The subject is simply one or more of the parameters to some of the available functions.**
- **Context can also contain the "current" function which is usually presented as a tool.**

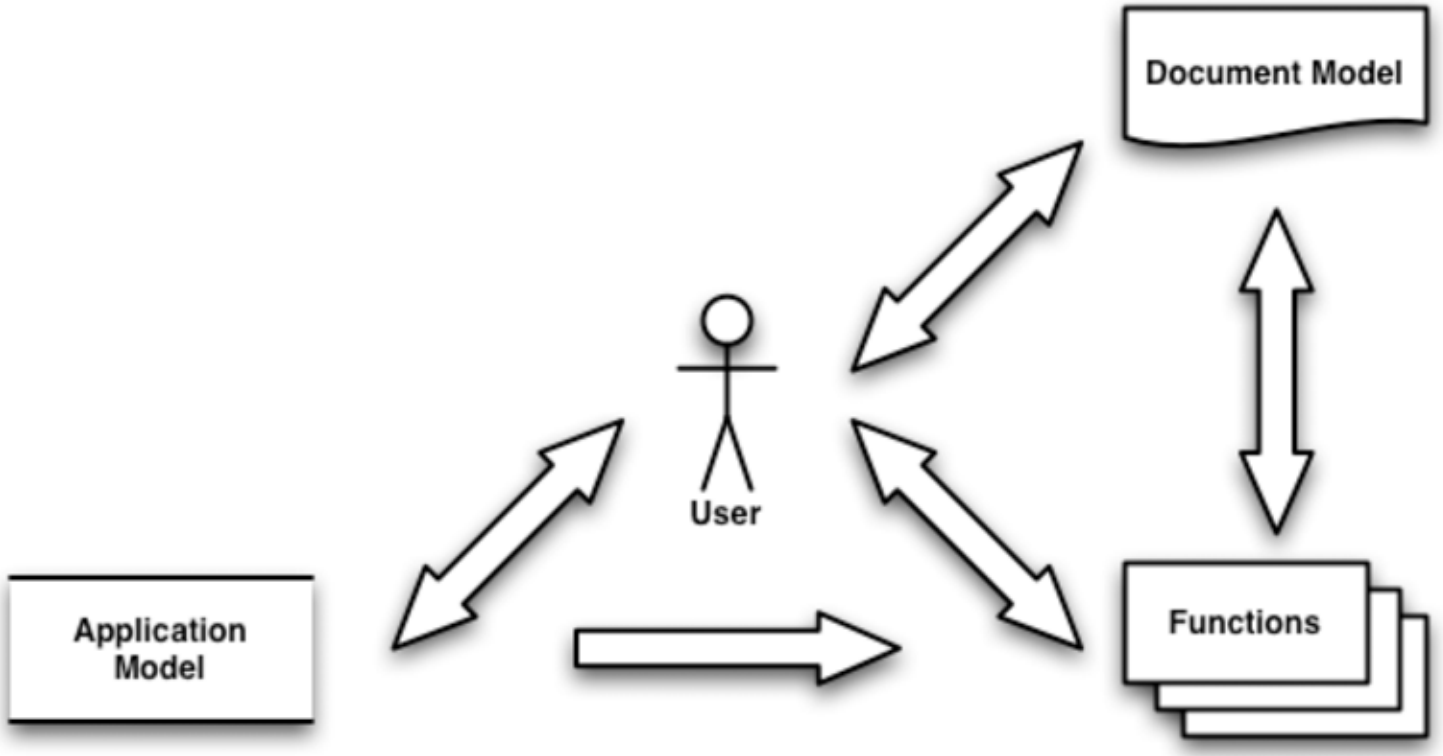
Example - Text Editor

- **Context**
  - **Current Document**
    - **Provides destination**
  - **Selection**
    - **Provides a range; an empty range denotes a location**



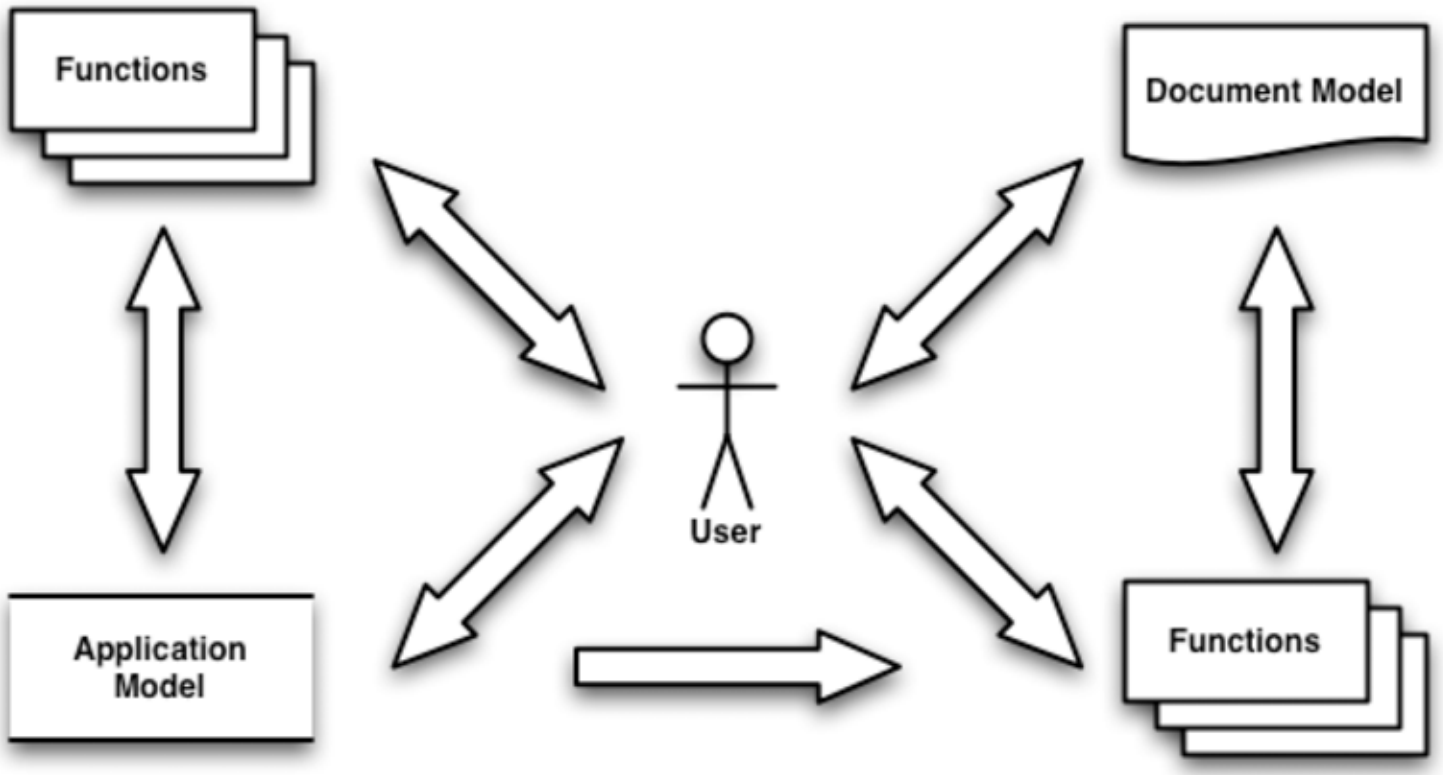
Adobe

Design Space



Adobe

Design Space



Adobe

Photoshop Context

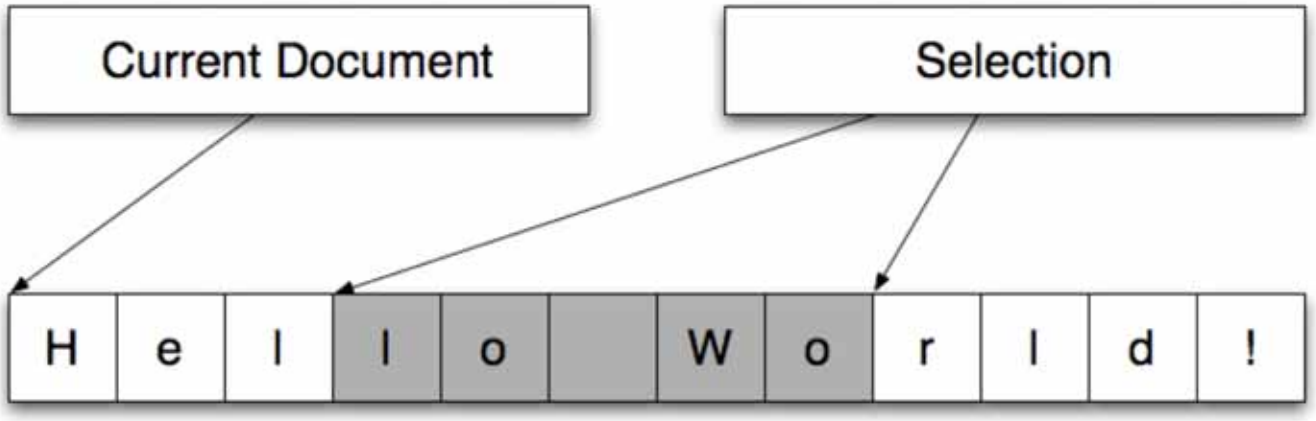
- Sequence of Documents
- Current Document
  - Current Layer
    - Current Channel
- Current Tool

Constraining Input

- **Only allow the user to select from valid options.**
  - **Functions can only be chosen if their preconditions can be satisfied by the subject.**
  - **Parameter values can only be set if they satisfy preconditions and currently contribute to the result.**

Example - Text Editor

- **The selection is defaulted to the beginning of the current document. The selection can only be changed to be valid within the document.**
- **An "Erase" command is enabled if there is a current document and a non-empty selection.**
- **An "Insert" command is enabled if there is a current document and an empty selection (note that location can be assumed to be valid).**



Adobe

Constraints in Photoshop

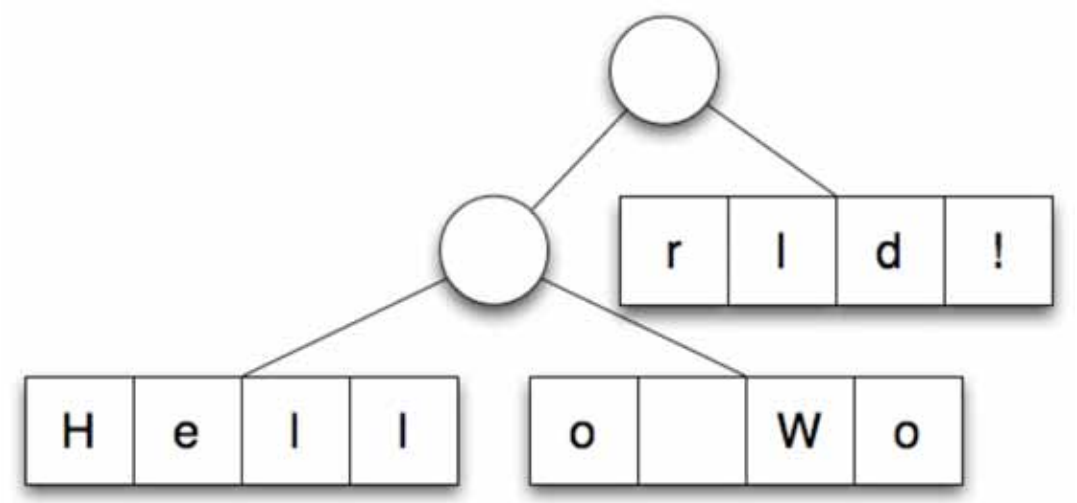
- **Commands can be enabled or disabled based on the current subject, or any attribute of the subject. For example, color space, color depth, number of channels**
- **Dialogs are a form of constraining the interface by focusing on the parameters to a single function**
- **Widgets may be disabled based on validity or potential contribution**

Interactive Assistance

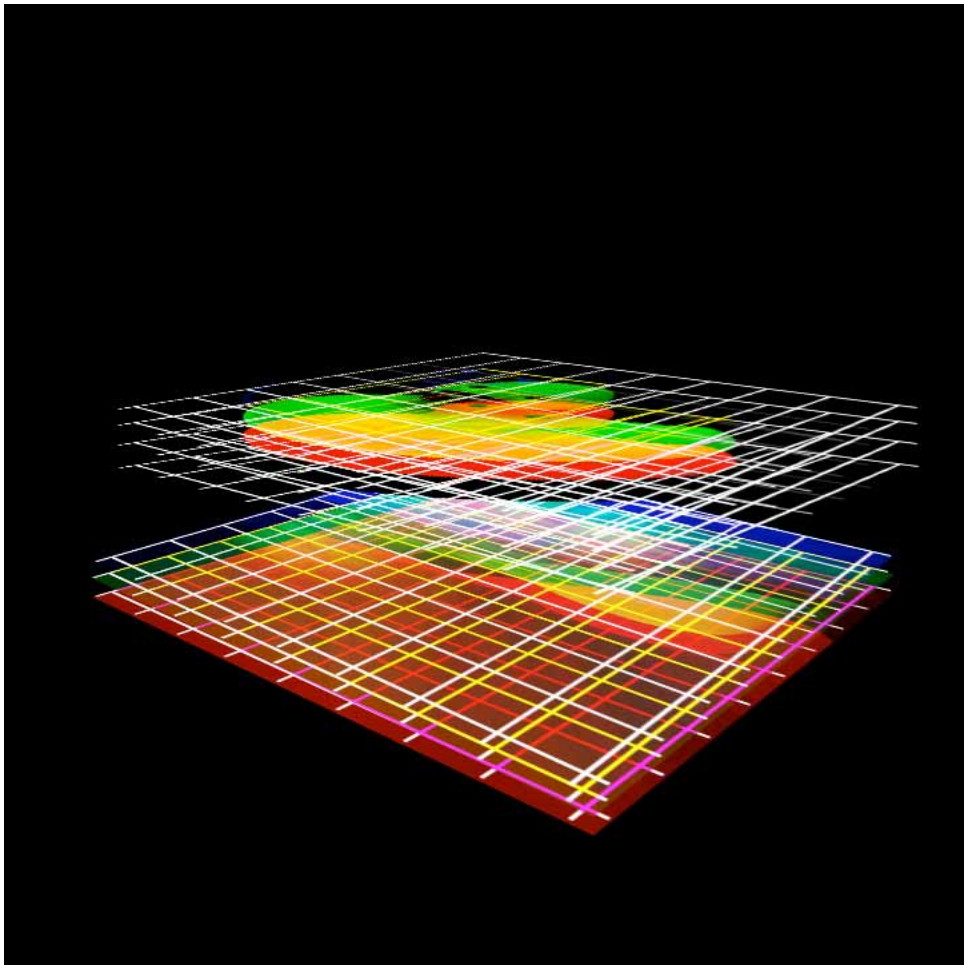
- **Tracking:** ≈1/30 s
- **Registration:** ≈1/5 s
- **Confirmation:** ≈1 s

Example - Text Editor

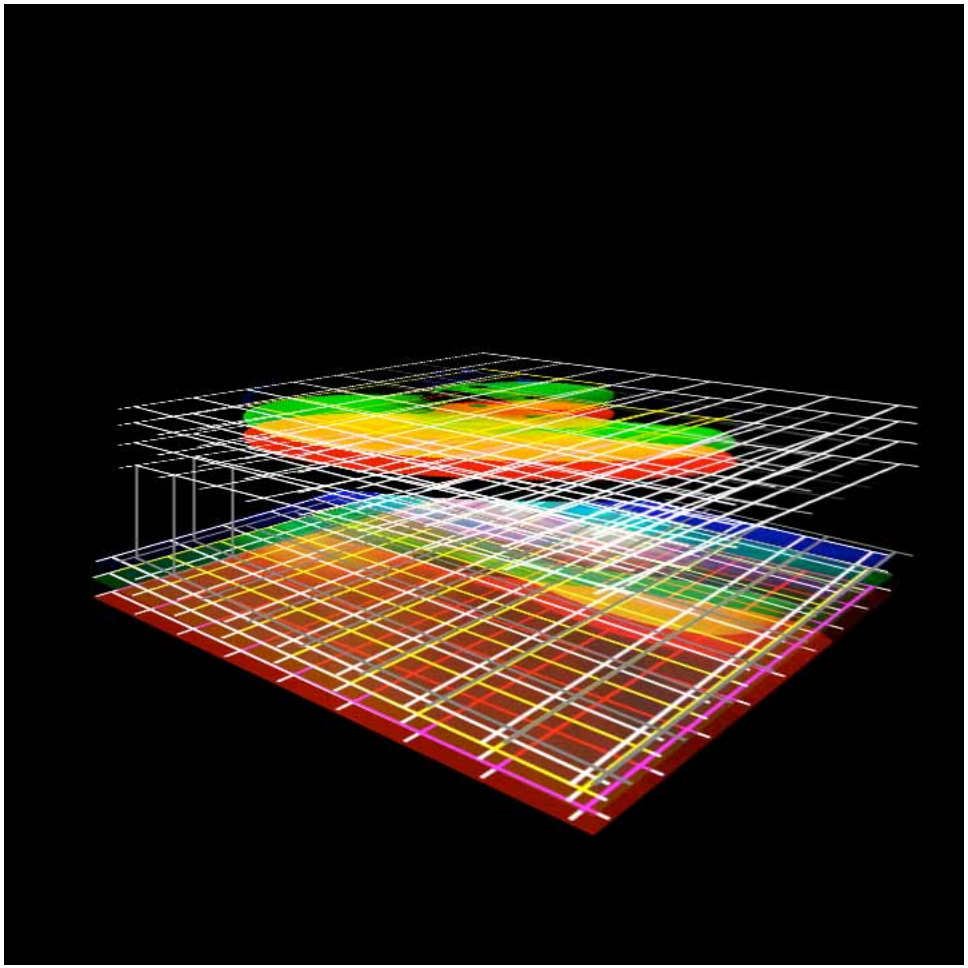
- **Need to be able to set the selection in "constant" time**
  - **This would imply a vector data structure**
- **Also need constant time insert and erase**
  - **This would imply a list data structure**
- **Solution: a more complex data structure such as a rope**



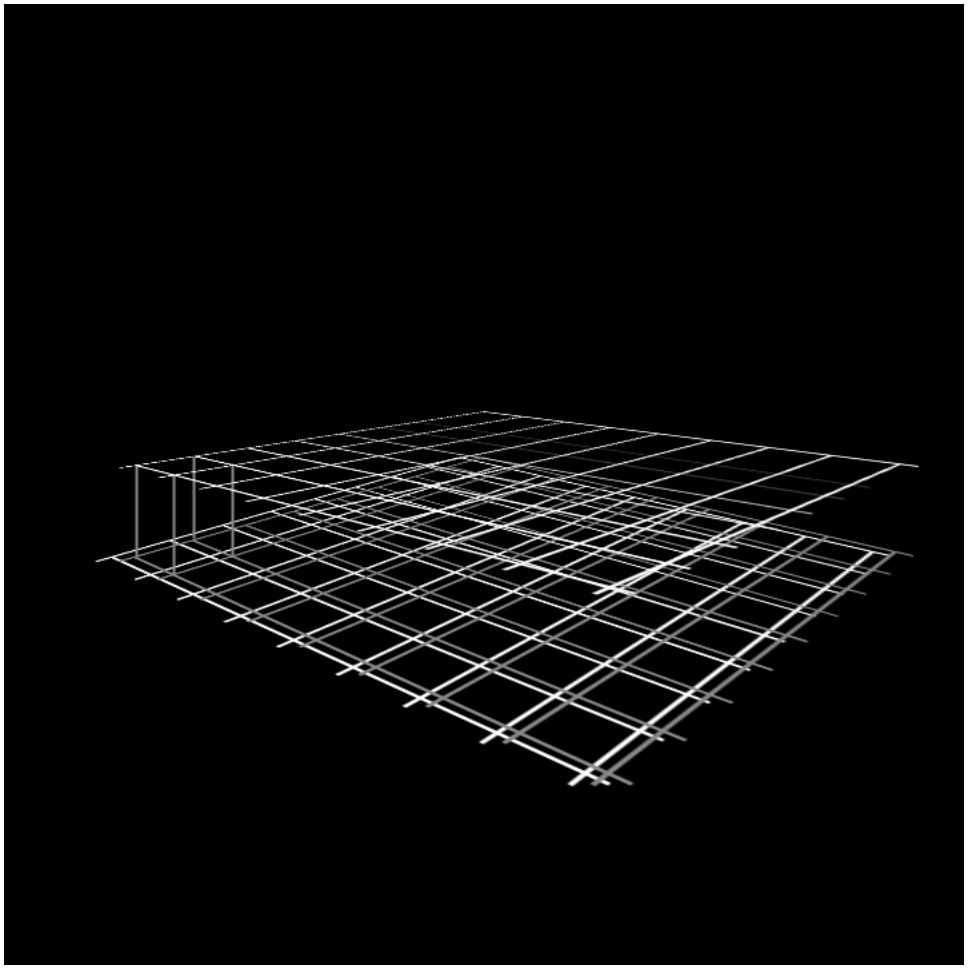
Tiles



Tile Alignment

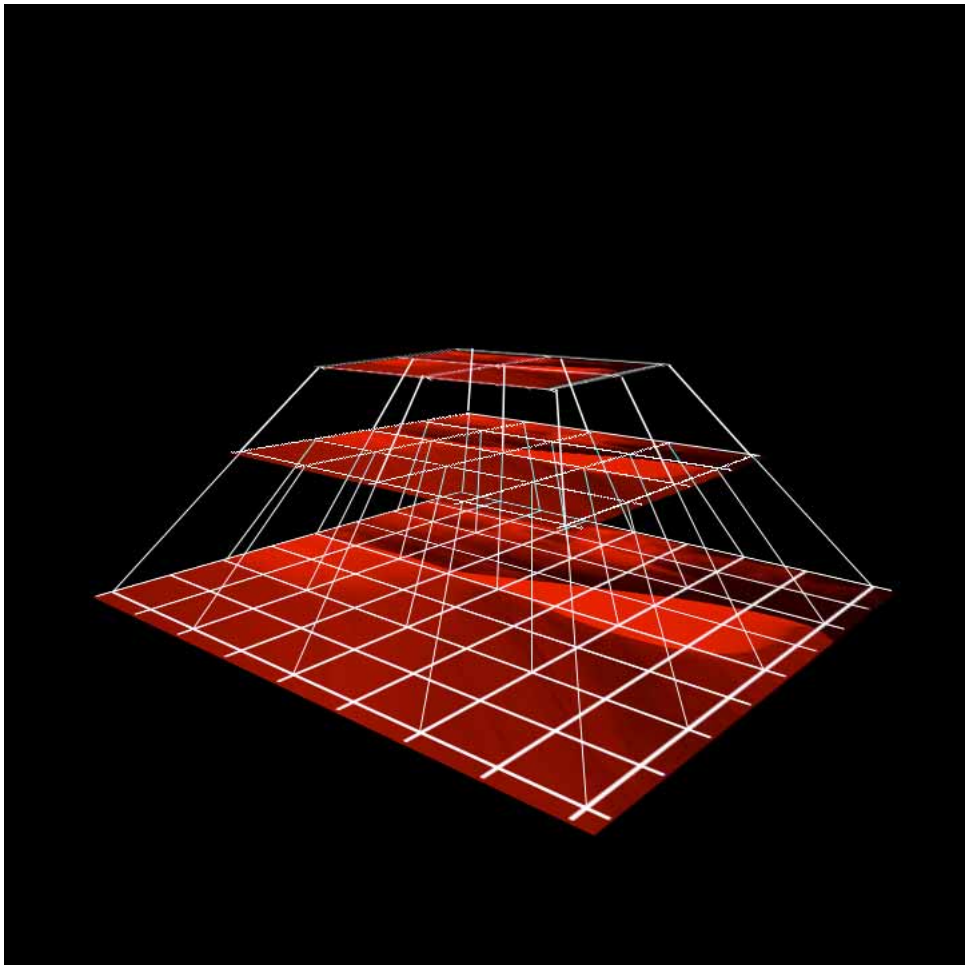


Tile Alignment



Adobe

MIP Mapping

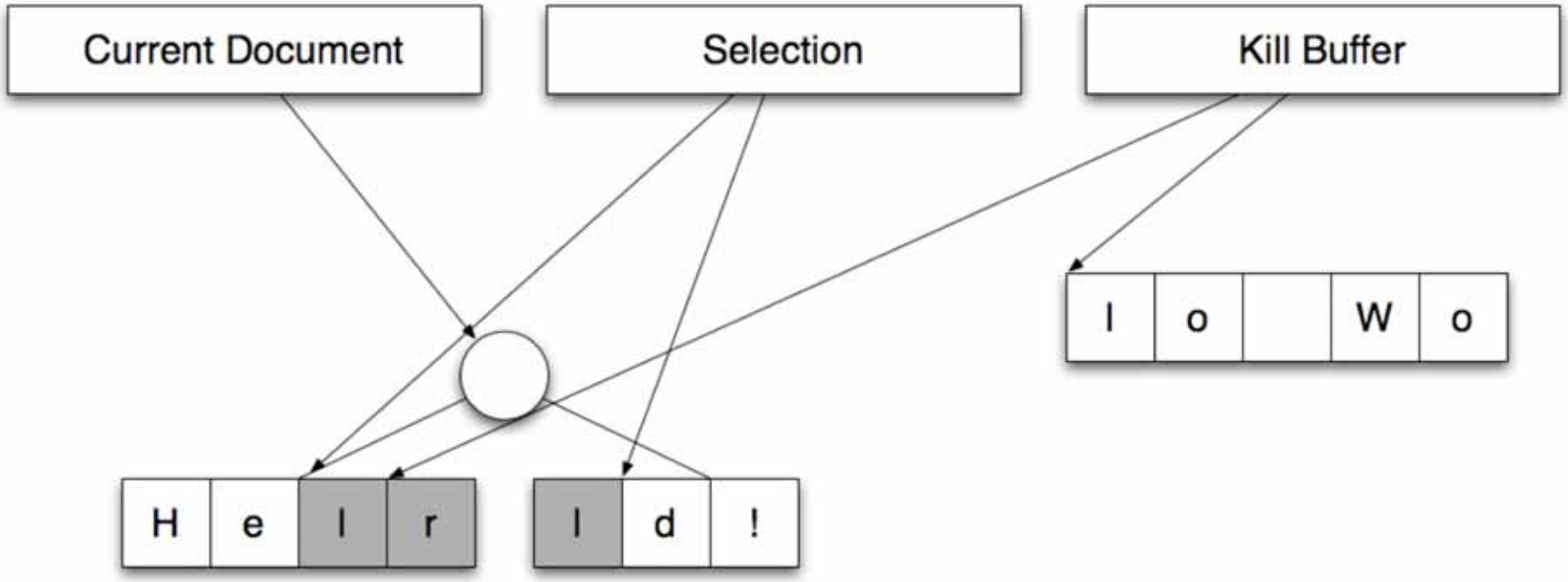


Prediction

- **Allow the user to state the request in terms of the desired result or to preview the result. This is accomplished by modeling the post-conditions of a functions**
- **Undo, Preview, Non-Destructive Editing and "Direct Manipulation" are all forms of predictive UI that completely model post-conditions**

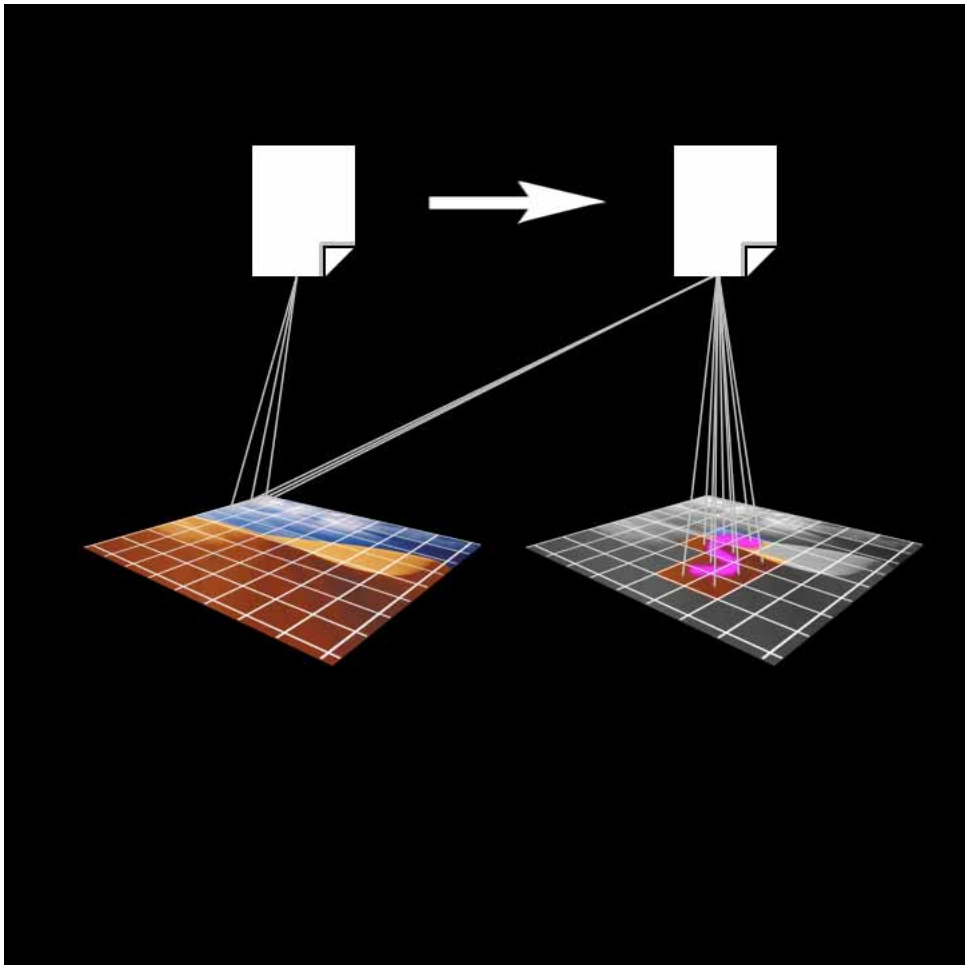
Example - Text Edit

- **To implement Undo we will need a "kill buffer" where we can store text that was erased and where it was erased from**
- **We will also need to be able to store the range of text that was last inserted**



Adobe

Photoshop History



Adobe

Demo

Scripting as User Interface

- **Same goals as a visual user interface**
  - **assistance sets scripting apart from an API**
  - **Less emphasis on interactive but still important as scripts often complement the visual interface**
- **Document model and functions remain intact**
  - **Application model and functions may vary**
- **Prediction and Constraints are critical**
  - **Defines scripting interface**
- **Recording - capturing contributing values. Related to prediction and modeling post conditions.**

Teach & Learn Core Computer Science

- **Algorithms**
  - **Including pre- and post- conditions and complexity**
- **Data Structures**
  - **Design and tradeoffs**
- **Computer Architecture**
  - **Absolute performance is as important as complexity**
- **Algebra**
  - **Software is defined on algebraic structures**

Links and References

- **Adobe Software Technology Lab: http://stlab.adobe.com/**
- **More from the Lab:**
  - **http://www.stepanovpapers.com/eop/lecture_all.pdf**

- **H.-J. Boehm, R. Atkinson, and M. Plass, "Ropes: an Alternative to Strings", *Software Practice and Experience* 25(12):1315, 1995.**

- **Thanks to Russell Williams for Photoshop code statistics.**

Revolutionizing

how the world engages with ideas and information

Adobe