Augmenting Reality via Client/Cloud Platforms

Avideh Zakhor
www-video.eecs.berkeley.edu/~avz
avz@eecs.berkeley.edu

Video and Image Processing Lab
University of California, Berkeley
Acknowledgements

- **Staff member:**
  - John Kua

- **Graduate students:**
  - Jerry Zhang, Aaron Hallquist, Matt Carlberg, Nick Corso

- **Undergraduate students:**
  - Eric Liang, Eric Tzeng, Jacky Chen, George Chen, Tim Liu

- **Earthmine Inc.:**
  - Jimmy Wang, John Ristevski
Outline

- What is Augmented Reality (AR)?
- Why now?
- Current examples and apps
  - Image based localization for AR apps
  - Indoor and outdoor
- Future directions of research
What is Augmented Reality?

- Enhance or augment real/actual world to create a more satisfying user experience/perception:

- Joining of virtual and actual reality

- “The real world is way too boring for many people. By making the real world a playground for the virtual world, we can make the real world much more interesting.” - Daniel Sánchez-Crespo, Novarama
AR Helps with Questions like ....

What is this?

Who are these people?
Familiar Real World Examples of AR

Heads up Display

Sports Broadcasting
Actual Applications...

Word Lens: Real time Translation; OCR

Dish-pointer for Satellite

Use GPS & orientation sensor

Yelp: Local reviews

Layar: Generalized AR platform
Google Goggles

- Google Goggle uses imagery for visual search, but:
  - Works well with famous landmarks
  - Doesn’t generalize to “typical streets”
- Most AR apps today leave a lot to be desired ….
Why now?

- Prevalence of mobile/handheld devices e.g. smart phones with lots of sensors:
  - Cameras
  - Coarse orientation measurement sensors:
    - Landscape vs. portrait on i-Phone
  - Coarse GPS
  - Coarse accelerometers
    - Game applications

1995  2011
Why now? (2)

- CPU cycles are more abundant and cheaper than ever
  - Cloud computing
- Wireless networks are getting faster
- Recognition performance improving by leaps and bounds in the last 6 years
Underlying technologies for AR...

- Suite of sensors to sense/recognize the environment & to localize user:
  - Camera
  - GPS & orientation sensors
- Algorithms to process sensor data ➔ signal/image processing, vision, recognition, ...
- Databases to look up meta data associated with user’s environment ➔ cloud storage
- Networks to communicate meta data to the user ➔ intermittent connectivity
- Present the data to the user ➔ User interface, rendering, visualization
Localization & Tracking

- Google's Schmidt: Location, mobile ads will revolutionize commerce
  Mobile World Congress, Barcelona, Spain, Feb. 14th, 2010;
  “A billion dollar business right in front of us.”

- Localization:
  - Means position and orientation
  - Indoor and outdoor

- Using GPS for outdoors:
  - Does not provide orientation.
  - Not accurate for most AR apps
    - Even differential GPS not accurate enough
    - Need pixel level accuracy
  - GPS satellites not always visible to mobile
    - Need to see three satellites
    - Urban environments with tall buildings, e.g. Manhattan
  - How about cell tower triangulation?
Cell Tower Triangulation

- **Mandate by FCC:**
  - 911 emergency services
  - Law enforcement
  - 67% of phones must be localized within 50 meters
  - 95% within 150 meters
How about WiFi?

- Dense urban environments with tall buildings likely to have great WiFi coverage:
  - Accuracy not large enough for AR applications
  - Privacy issues
Image Data Bases (dB) & Localization

- Drawbacks of existing approaches:
  - Not accurate to pixel level
  - Do not provide orientation

- Use images to overlay info/tags/meta-data on viewfinders to achieve pixel level accuracy
  - Image based localization

- Need Large image databases:
  - Street View from Google,
  - Bing maps from Microsoft,
  - Earthmine, etc
Mass scale image acquisition systems

Google Street View

Earthmine
Google StreetView Picture for the Intersection of Hearst and LaLoma, Berkeley, CA

Google Earth 3D model of NY
Ground-Based Modeling

“Drive-by Scanning”
2x2D laser scanners + camera mounted on a truck

Airborne Modeling:
Laser scanners and cameras on planes and helicopters

Flythrough rendering
Walkthrough rendering

Frueh & Zakhor
2003, 2004, 2005
Satellite View with 3D Buildings superimposed (UCB Modeling Tool)
How to use Image Database to “Localize”

- Compute features of images in the dB and the query image
  - Bag of words model

- Train a vocabulary tree or K-D tree using all the features from dB images
  - Quantizes features that are close to each other in the same “bin”

- Input the features of the query image to the “tree”

- Score for each dB image:
  - number of “matched features” to the query

- Find largest score dB image


Performance degrades with the size of images in the database
Divide and Conquer → Scalable

- Divide a large geographic area into overlapping circular “cells”
  - Centered at vertices of hexagonal lattice
  - Similar to “handoff” in wireless carriers
- Each cell has its own k-d tree
- Coarse location reported by cell phone:
  - GPS or cell tower triangulation
  - Actual location is within *ambiguity circle* centered around reported location
  - Probability distribution function from FCC

![Diagram of hexagonal lattice with overlapping circular cells and an ambiguity circle centered around a reported location.]
Optimal Geometry for the Cells

- Cell radius $R$;
- Ambiguity circle radius $G$;
- Distance between center of cells: $D \geq \text{Overlap}$
- To ensure entire ambiguity circle lies inside at least ONE cell:

$$d \leq \sqrt{3}(r - g)$$

- Can get away with just ONE cell search
Combine Results of Multiple Cells

- Assume \( D = R \)
- Each image is in
  - 4 cells if in “petal”, 3 cells if not in “petal”
  - With zero ambiguity, can combine 3 to 4 cells to improve results
- Ambiguity circle can overlap with 3 to 9 cell
  - Search all cells Amb. Cir. intersects with, even if matched image in only 3 to 4 cells.

- Combine the scores of dB images from various cells

Query ~ location

- Find intersection with all cells
- SIFT Features

Parallel K-D tree search & combination
Example: no location ambiguity
Example with ambiguity
Filtering to Improve Results

**Query**

~ location

- Find intersection with all cells
- SIFT features

Parallel K-D tree search & combination

**Filtering**

Filtering (1): Geometric verification (GV) via RANSAC to eliminate erroneous feature matches

Filtering (2): Compute the ratio between closest feature match & second closest feature match
Filtering (3): Machine Learning

- Train a Naïve Bayes Classifier:
  - Training set: 65 query image; each with on average 100 “candidate” matches;
  - Extract distance \( d \) from reported location & geometrically verified SIFT feature votes \( v \)
  - Generate the prior and conditional distributions \( p(m) \), \( p(d|m) \), and \( p(v|m) \); \( m = \) match;

- Test the classifier on new data by extracting votes & distance

Classifier predicts the probability \( p(m=1|d,v) \) that a candidate image is a match
- Rank order dB image set
- Confidence level in each dB image
Match Confidence is a good indicator of Match Performance

Match Performance vs Match Confidence

Based on confidence of the best match, can ask the user to re-take the query image, if needed
Earthmine Data Set: Panoramic Locations
Data Sources

- **Image Database:**
  - ~2000, 360 degree panoramic images of downtown Berkeley
  - Processed into ~12000 geo-tagged 768x512 "street-view" images
  - One square kilometer
  - 25 cells of radius 236 m
  - ~1500 images per cell

- **Query Set**
  - Camera SLR Nikon camera D40x w/ 18-55mm lens:
    - Sets 1 and 2: wide angle
    - Set 3: varied focal length
      - Wide angle, zoom, normal
    - ~90 landscape images per set
  - Cell phone camera
    - HTC Droid Incredible
    - 8 megapixel camera, autofocus, focal length 4.92mm
    - ~110 portrait images per set
  - Geo-tag images: GPS on cell phone:
    - +/- 10 meter accuracy ➔ too fine
    - Emulate errors of up to 100 to 200 meters
**Experimental Setup and Results**

**Causes of Failure:**
- Query pictures taken close-up often with shadows
- Heavily obscured by tree branches
- Not a correct pose match in the db
- Matched common objects
Comparisons with Existing Approaches

- **Pollefeys et. al. 2010**
  - Uses Earthmine database
  - San Francisco, not downtown Berkeley
  - ~30000 images in database
  - Good performance if trained and tested on Earthmine
  - Much lower performance than our system for actual cell phone

<table>
<thead>
<tr>
<th></th>
<th>Affine</th>
<th>Masked</th>
<th>Rectified</th>
<th>Upright</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earthmine</td>
<td>84.3%</td>
<td>83.0%</td>
<td>82.6%</td>
<td>85.0%</td>
</tr>
<tr>
<td>Navteq</td>
<td>33.9%</td>
<td>26.3%</td>
<td>25.2%</td>
<td>35.7%</td>
</tr>
<tr>
<td>Cellphone</td>
<td>30.2%</td>
<td>23.2%</td>
<td>25.2%</td>
<td>32.1%</td>
</tr>
</tbody>
</table>

- **Girod et. al. 2008**
  - Discretizes user location on-the-fly
    - 30m x 30m cells/loxels ➔ 20 times smaller cell size
    - Assumes near perfect GPS localization
  - Generates kd-tree on the client from 9 loxels
Annotating Query Image

Earthmine Tagger: User tags panoramas
Tags are converted to 3D locations in space
Project all tags in vicinity of database image onto database image plane

- Use location/orientation info of image dB
- Query image might have more Buildings/landmarks than dB image
Transfer Tags onto Query Image
Transfer Tags onto Query Image (2)
Future Technical Challenges

- Optimum division of the computation between cloud and client
  - Battery drainage considerations
  - CPU asymmetry between cloud and client
  - Communication cost between cloud and client
    - Cloud processing for one time image based localization
      - Takes 6 second on a server:
        - 2 seconds for finding SIFT features
        - 2 seconds to do k-D tree processing
        - 2 seconds for combining & filtering
        - Assumes compressed JPEG image sent to the cloud

- Tracking the user and updating the tags:
  - Real time; interactivity
  - Initial localization at cloud; update at the handheld
Model Update via User Generated Image/Video Content → crowdsourcing

- Laser
- Lidar
- Radar
- Camera
- GPS
- Maps
- Gyroscope

3D model construction with texture

Mobiles with cameras
- Up to date databases
- Selectively update database
- Popular places updated more frequently
Indoor AR applications

- Why indoors? Shopping centers, airports,
  - Holy grail of mobile advertising & location based services
- No GPS:
  - No easy way to come up with coarse localization for AR
  - Automatic 3D modeling of indoors is hard ➔ research area
UC Berkeley: First in 3D Indoor Modeling

- Use a human backpack equipped with sensors to automatically generate 3D photorealistic textured models of indoor environments.
Key Step in 3D Model Construction: Loop Closure

- Loop closure (LC):
  - Revisiting same location
  - Reduces error in 3D model construction
  - Use Camera to detect LC automatically

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Path length</th>
<th>Average Position Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>68.73 m</td>
<td>0.66 m</td>
</tr>
<tr>
<td>2</td>
<td>142.03 m</td>
<td>0.35 m</td>
</tr>
<tr>
<td>3</td>
<td>46.28 m</td>
<td>0.58 m</td>
</tr>
<tr>
<td>4</td>
<td>142.03 m</td>
<td>0.43 m</td>
</tr>
</tbody>
</table>

Set 1

Set 2

Set 3

Set 4

Loop closures
Automatic Image Based Loop Closures (AIBLC)

- Same approach for AIBLC for indoor modeling can be applied to indoor AR localization
  - OK not to have GPS or any other coarse indoor localization

- Details:
    - Generate rank ordered list of candidate image pairs
    - Prune the list via "key point matching"

- Upshot: Same basic approaches of outdoor AR localization can now be applied to indoors
Augmented Reality in 2020

- Almost certainly, many more AR apps on cell phones:
  - Mobile advertising
  - Location based service
- Most likely, 3D AR apps with compelling user experience:
  - Gaming and entertainment
- Ultimate goal: blur the line between real and virtual
Summary and Conclusions

- AR no longer the technology of the future;
  - All key technological ingredients are available here today
  - Sensor equipped cell phones, fast networks, image recognition, user interface, databases, cheap CPU

- Only a matter of putting these together in the right way to truly enhance user experience