



Computer Science in Practice

67864

<http://www.cs.huji.ac.il/~csip>



Why?

- Undergraduate curriculum is lacking
- Large gaps between Theory and Practice
- Exposure to some “applied” fields
- Exposure to numerical techniques



What?

- Discuss selected topics from three areas:
 - Machine Learning
 - Computer Vision
 - Computer Graphics
- Common denominator: solutions use tools from numerical linear algebra
- Not a replacement for courses in the above areas
- Mostly intended for students not specializing in one of these areas



Topics

- Google's PageRank and the Power Method
- Machine Learning (4 weeks):
 - Supervised learning: regression and the pseudo-inverse
 - Supervised classification (Fisher Linear Discriminant)
 - Unsupervised learning: dimensionality reduction using PCA



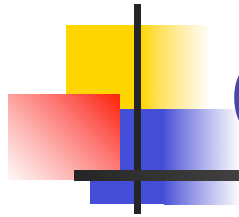
Topics

- Computer Vision (4 weeks):
 - Spectral segmentation
 - Optical flow computation (Lucas-Kanade)
 - Structure from motion (Tomasi-Kanade)
- Computer Graphics (4 weeks):
 - High Dynamic Range compression
 - Solving large, sparse linear systems
 - Graphical image manipulation



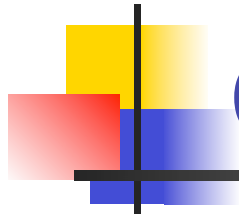
Requirements

- Programming assignments in Octave
- Theoretical exercises
- Exam



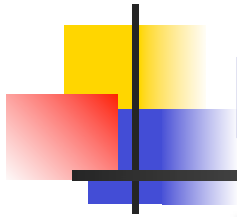
Google's PageRank Algorithm

- A living example of “computer science in practice”
- History:
 - 1993: Mosaic, the first popular graphical browser for the WWW, is released.
 - Internet search, as we know it today, does not exist:
 - Archie (1990), the first search engine designed to search FTP archives by filename.
 - Gopher (1991), a distributed document search and retrieval network protocol.



Google's PageRank Algorithm

- History (continued):
 - WebCrawler (1994) was the first engine to allow searching for any word in any webpage
 - Directories and portals, e.g., Yahoo! (1994)
 - Search engines: Excite, AltaVista (1995)
 - 13 million queries per day
 - Google
 - 1998 --- 25 million web pages indexed
 - 2005 --- 8 billion web pages indexed, 200 million queries per day



How to Order Search Results?

[Sign in](#)

Google [Web](#) [Images](#) [Video](#) [News](#) [Maps](#) [more »](#)

[Advanced Search](#)
[Preferences](#)

Web Results 1 - 10 of about 51,000 for **Dani Lischinski**. (0.02 seconds)

[Dani Lischinski - Home Page](#)
Dani Lischinski. School of Computer Science and Engineering · The Hebrew University of Jerusalem. Givat Ram, Jerusalem 91904, Israel ...
www.cs.huji.ac.il/~danix/ - 12k - [Cached](#) - [Similar pages](#)

[Gradient Domain HDR Compression](#)
Gradient Domain High Dynamic Range Compression. Raanan Fattal, **Dani Lischinski**, Michael Werman. Abstract. We present a new method for rendering high dynamic ...
www.cs.huji.ac.il/~danix/hdr/ - 3k - [Cached](#) - [Similar pages](#)
[[More results from www.cs.huji.ac.il](#)]

[DBLP: Dani Lischinski](#)
8 · Frederic H. Pighin, **Dani Lischinski**, David Salesin: Progressive Previewing of Ray-Traced Images Using Image Plane Discontinuity Meshing. ...
www.informatik.uni-trier.de/~ley/db/indices/a-tree/l/Lischinski:Dani.html - 32k - [Cached](#) - [Similar pages](#)

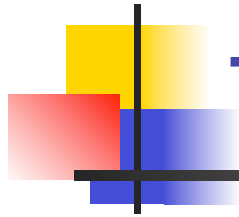
[Hierarchical image caching for accelerated walkthroughs of complex ...](#)
2 Bradford Chamberlain , Tony DeRose , **Dani Lischinski** , David Salesin , John Snyder, Fast rendering of complex environments using a spatial hierarchy, ...
portal.acm.org/citation.cfm?id=237209&coll=portal&dl=ACM - [Similar pages](#)

[Synthesizing realistic facial expressions from photographs](#)
Dani Lischinski. **Dani Lischinski**. Frédéric Pighin. Frederic Pighin. David H. Salesin ... **Dani Lischinski**. Nuria Oliver. Victor Ostromoukhov. Yunhe Pan ...
portal.acm.org/citation.cfm?coll=GUIDE&dl=GUIDE&id=280825 - [Similar pages](#)
[[More results from portal.acm.org](#)]



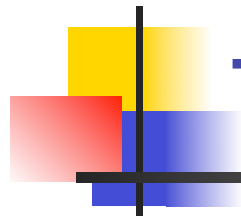
Challenges

- Importance of pages can be subjective
 - Diversity of pages is extremely high
 - Searches initiated by inexperienced users
 - “Engineered” pages
-
- **Approach:** derive page importance from the link structure of the Web.



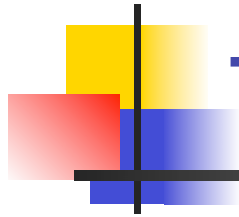
The PageRank Idea

- Important pages will have many “incoming” links
- An incoming link from an important page is worth more than others
- Conclusion: link importance is a recursive notion.
- What is the appropriate mathematical model?



The “Random Surfer” model

- Random surfer:
 - At time t , the random surfer is visiting page P_t
 - With probability $1-\varepsilon$ follow one of the outgoing links
 - With probability ε proceed to a random URL
- The importance of a page i is defined as the probability that the “random surfer” is visiting the page at any given time (stationary probability).



The "Random Surfer" model

- Transition graph (matrix):

$$A_{ij} = \Pr\left(x^{(t+1)} = i \mid x^{(t)} = j\right)$$

$$A_{ij} = \begin{cases} \frac{\varepsilon}{N} & \text{no link } j \rightarrow i \\ \frac{1-\varepsilon}{M_j} + \frac{\varepsilon}{N} & \text{link } j \rightarrow i \text{ exists} \end{cases}$$

- N – total number of pages, M_j – number of links on page j .

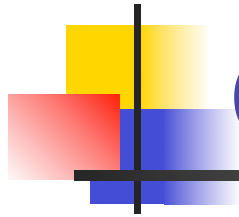


PageRank

- Let $p(i) = \text{PageRank}(i)$ be the probability of being at the i -th page at any given time.
- $p(i)$ may be expressed as:

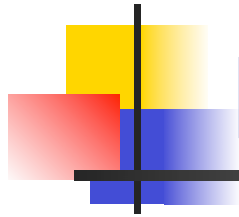
$$p(i) = \sum_{j=1}^N p(j) A_{ij} \quad \longrightarrow \quad p = A p$$

- Conclusion: the vector p is a right eigenvector of the transition matrix A , whose corresponding eigenvalue is 1.



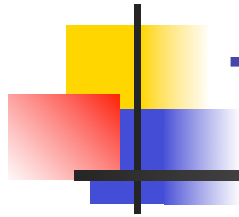
Computing the PageRank

- Idea 0: solve the linear system $(A - I)p = 0$
- Actually, this is a bad idea...
 - Huge matrix
 - Sparsity lacks structure
- Fortunately, there is a better way!



Perron-Frobenius Theorem

- Let A be a stochastic $n \times n$ matrix with positive entries ($a_{ij} > 0$). Then the following statements hold:
 - $\lambda = 1$ is a (real) simple eigenvalue of A , and all other eigenvalues λ' satisfy $|\lambda'| < 1$.
 - There exists a positive vector with entries summing to 1, which is the right eigenvector of A associated with the eigenvalue 1.



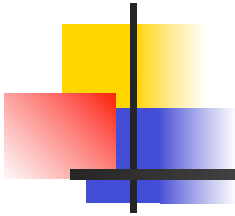
The Power Method

- Algorithm for computing the dominant eigenvector of A :

do $y \leftarrow Ax^{(t)}$

$$x^{(t+1)} \leftarrow \frac{y}{\|y\|}$$

until $\|x^{(t+1)} - x^{(t)}\| < \varepsilon$



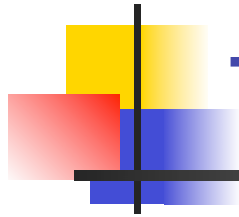
The Power Method: proof

$$A^t x^{(0)} = A^t \sum_{i=1}^n \alpha_i u_i = \sum_{i=1}^n \alpha_i A^t u_i = \sum_{i=1}^n \alpha_i \lambda_i^t u_i$$

$$= \alpha_1 \lambda_1^t u_1 + \sum_{i=2}^n \alpha_i \lambda_i^t u_i$$

$$= \alpha_1 \lambda_1^t \left(u_1 + \sum_{i=2}^n \frac{\alpha_i}{\alpha_1} \left(\frac{\lambda_i}{\lambda_1} \right)^t u_i \right)$$

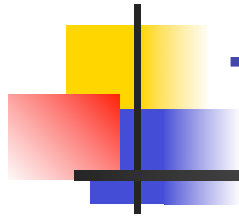
$$\sum_{i=2}^n \frac{\alpha_i}{\alpha_1} \left(\frac{\lambda_i}{\lambda_1} \right)^t u_i \xrightarrow{t \rightarrow \infty} 0$$



The Power Method: proof

- Recall that $x^{(t)}$ is normalized at each iteration:

$$\begin{aligned} x^{(t)} &= \frac{Ax^{(t-1)}}{\|Ax^{(t-1)}\|} = \frac{A^t x^{(0)}}{\|A^t x^{(0)}\|} \\ &\xrightarrow{t \rightarrow \infty} \frac{\alpha_1 \lambda_1^t u_1}{\|\alpha_1 \lambda_1^t u_1\|} = u_1 \end{aligned}$$



The Power Method: notes

- Convergence depends on the ratio $\frac{|\lambda_2|}{|\lambda_1|}$
- Obtaining the eigenvalue (for a unit eigenvector u):

$$u^T A u = u^T \lambda u = \lambda u^T u = \lambda$$