REU - Week 3 & 4

Kevin Kyyro

June 10th, 2010

Last Week

- Depth from defocus
- Lots of reading
  - “Depth from Diffusion” by Zhou, Cossairt, & Nayar
  - “Shape from Defocus via Diffusion” by Favaro, Soatto, Burger, & Osher
  - “3D Shape Estimation and Image Restoration” by Favaro & Soatto
  - More powerpoints and papers on DfDefocus and related programming
- Began test programming
- Began reading existing code

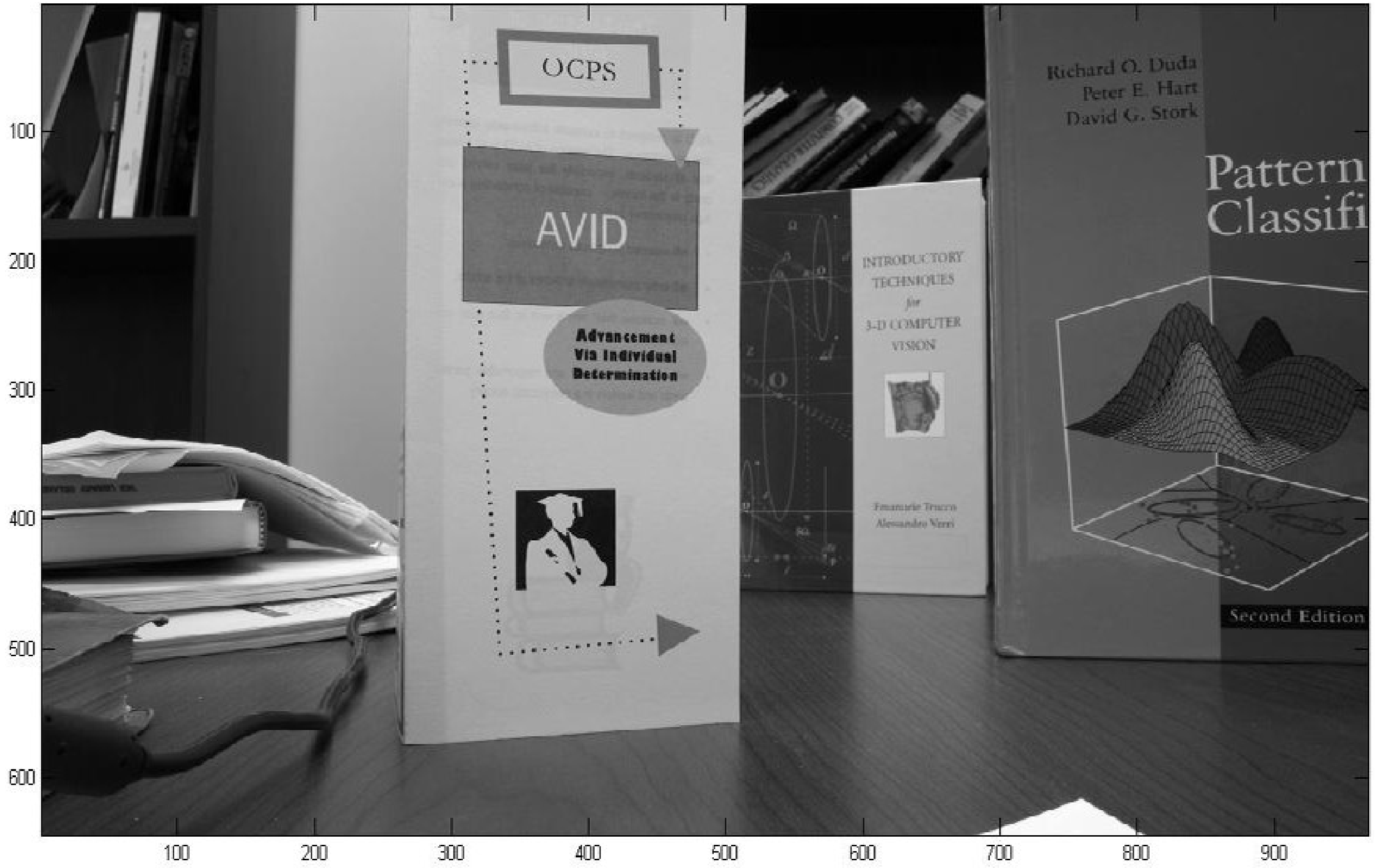
This Week

- More reading
  - Papers and code
- Finished test programming
  - Basic, intuitive implementation of depth from defocus
  - More on that next
- Brain storming
  - What are the limitations?
  - How to work around them?

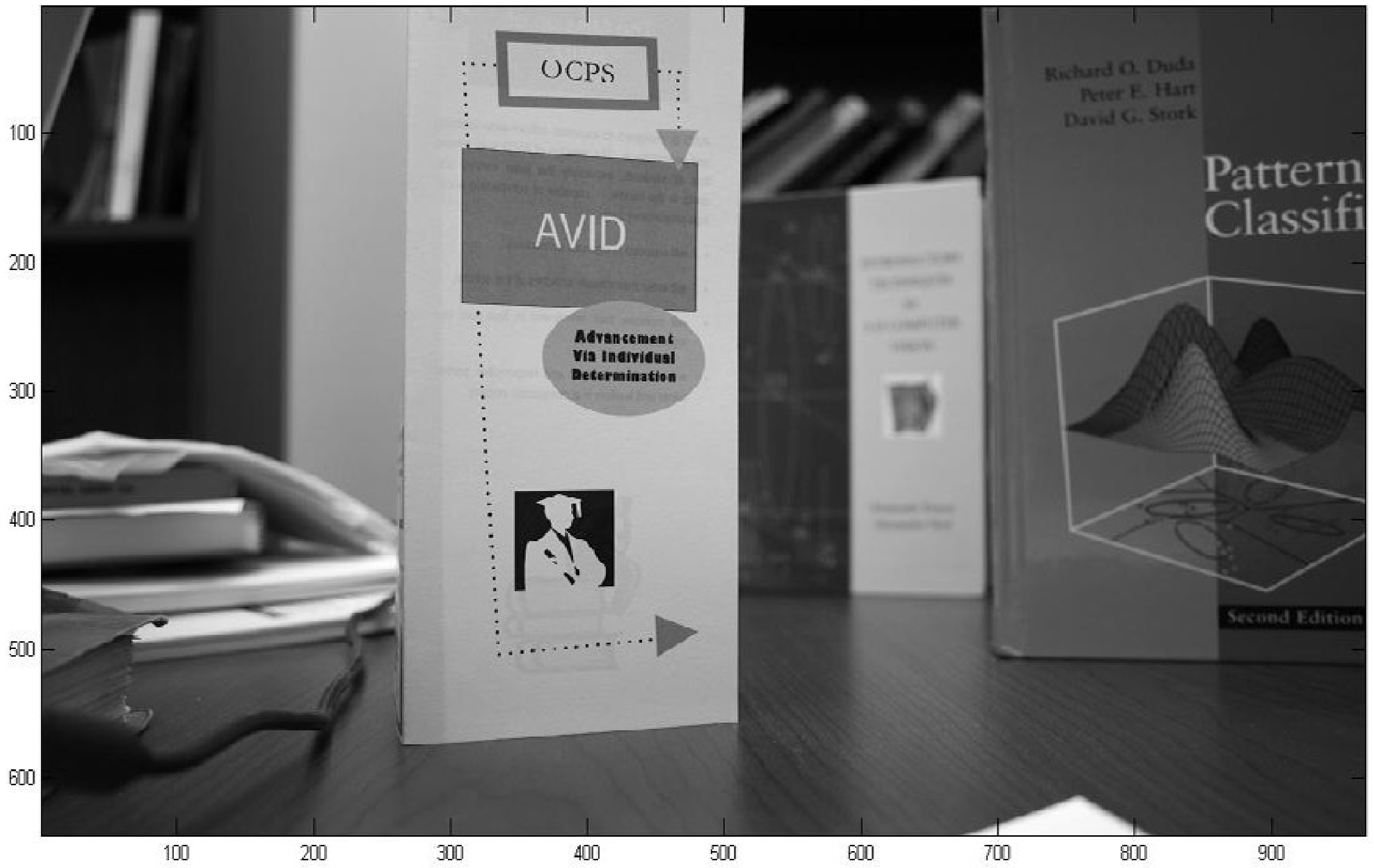
My Test

- Two images
  - One with deep depth of field
  - One with shallow depth of field
- Use a set of convolutions to estimate depth locally
- Essentially, reduces the problem to a grid of approximate coplanar patches (parallel to sensor)
- Decent results, but slow (at least in MATLAB)
- No image segmentation used, so depth only measured at edges

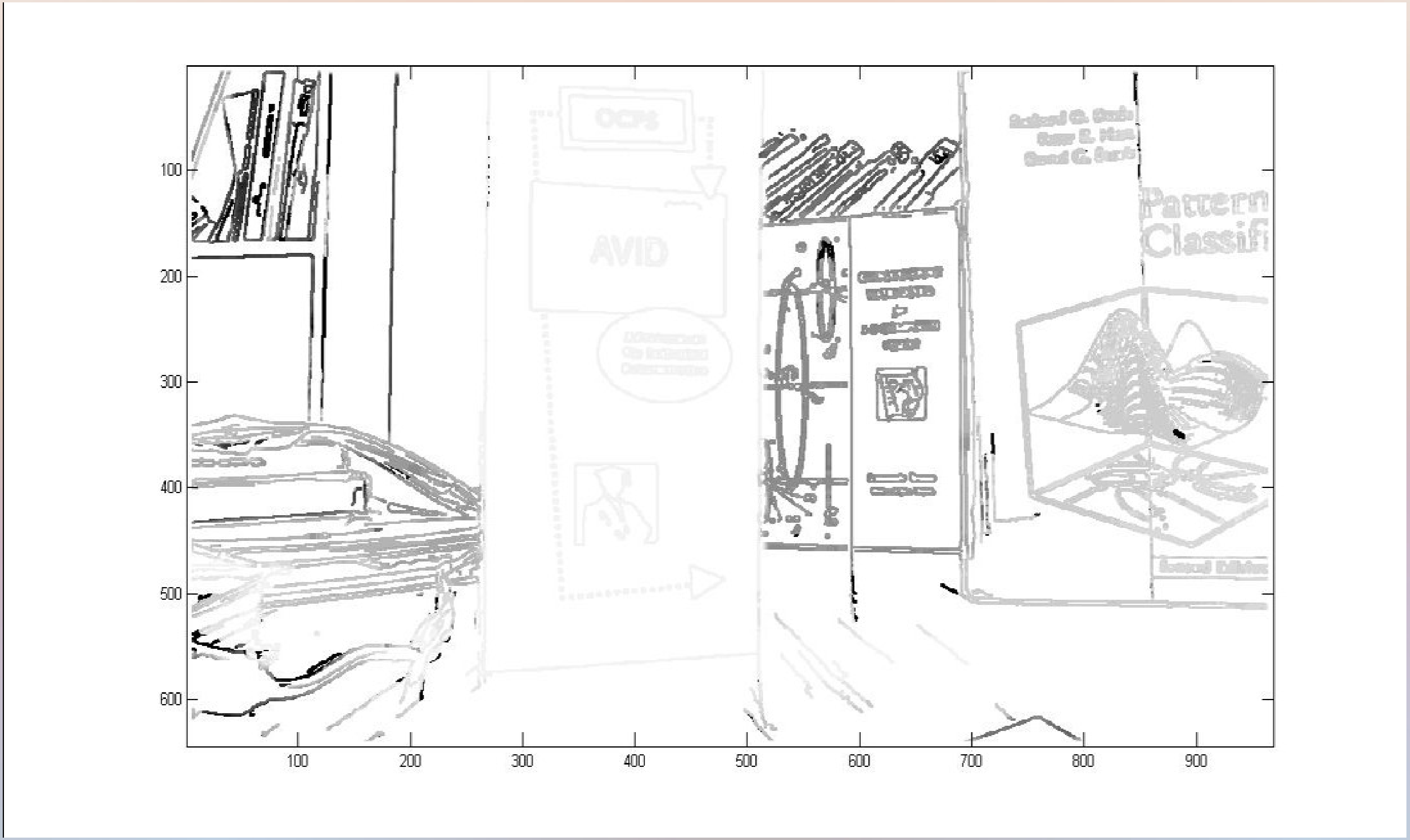
Deep DOF


100
200
300
400
500
600
700
800
900
OCPS
AVID
Advancement
Via Individual
Determination
INTRODUCTORY
TECHNIQUES
for
3-D COMPUTER
VISION
Emanuele Trucco
Alessandro Verri
Richard O. Duda
Peter E. Hart
David G. Stork
Pattern
Classifi
Second Edition

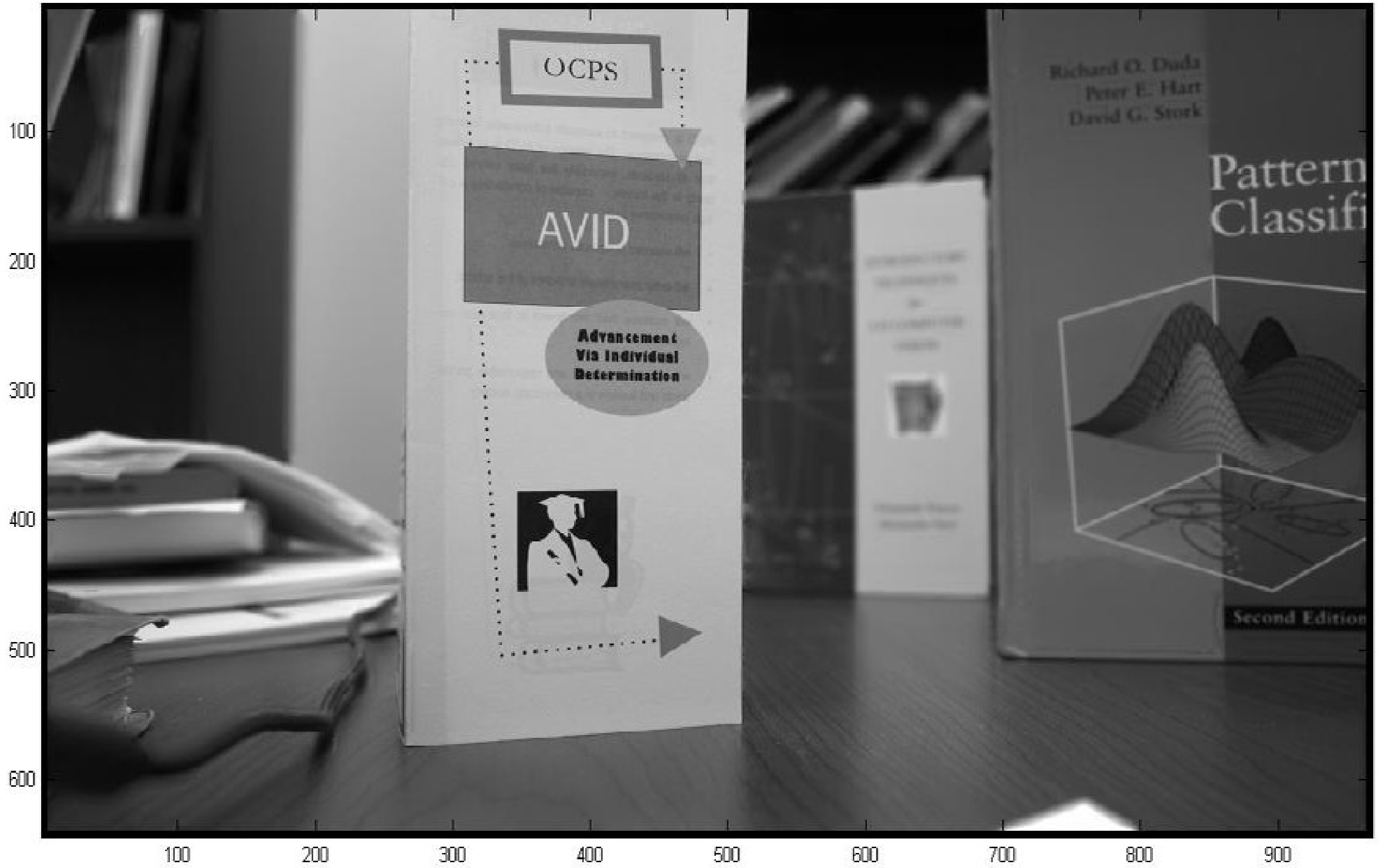
Shallow DOF


OCPS
AVID
Advancement
Via Individual
Determination
Richard O. Duda
Peter E. Hart
David G. Stork
Pattern
Classifi
Second Edition
100
200
300
400
500
600
100
200
300
400
500
600
700
800
900

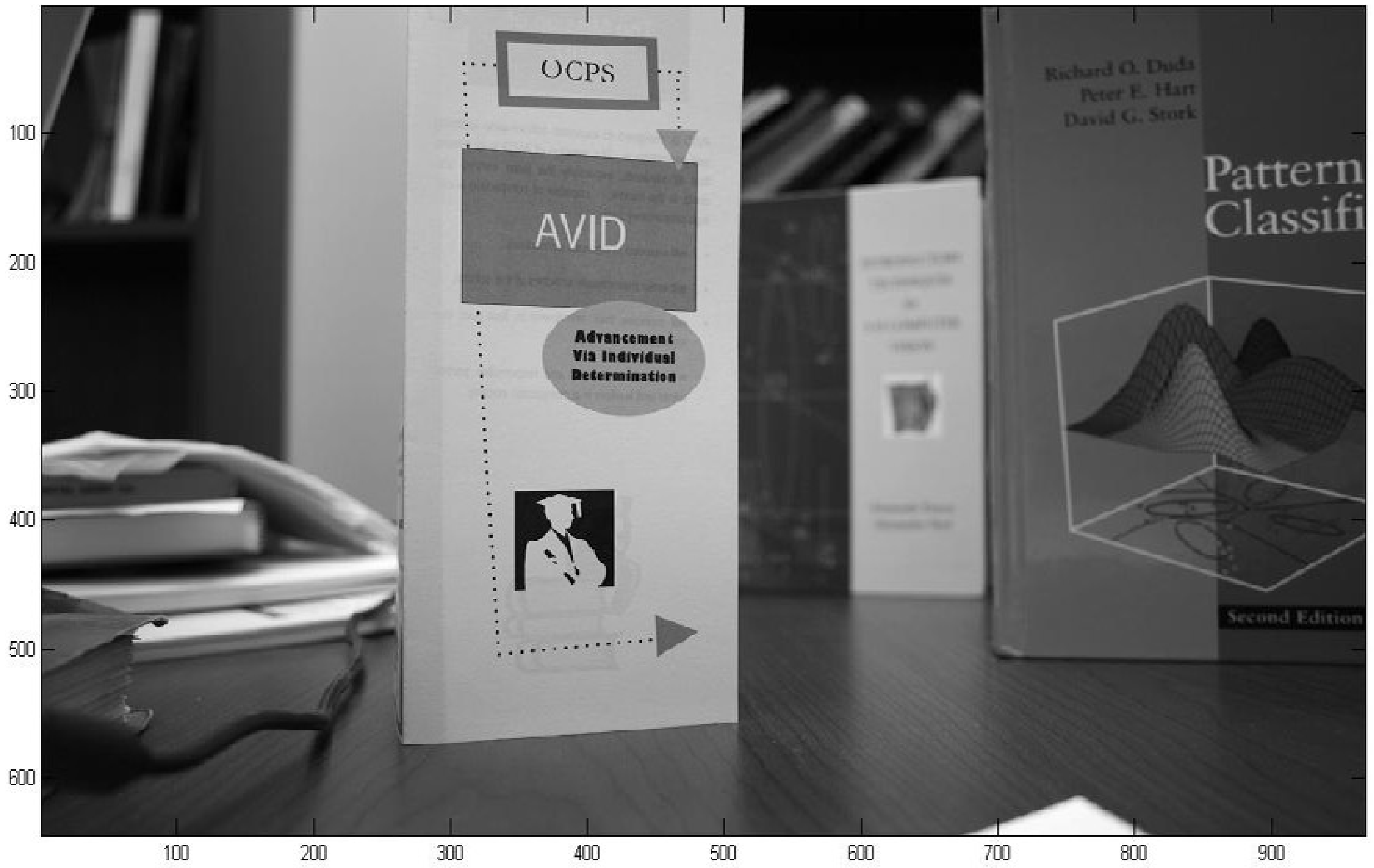
Depthmap


100
200
300
400
500
600
100
200
300
400
500
600
700
800
900
OCPS
AVID
Pattern
Classifi

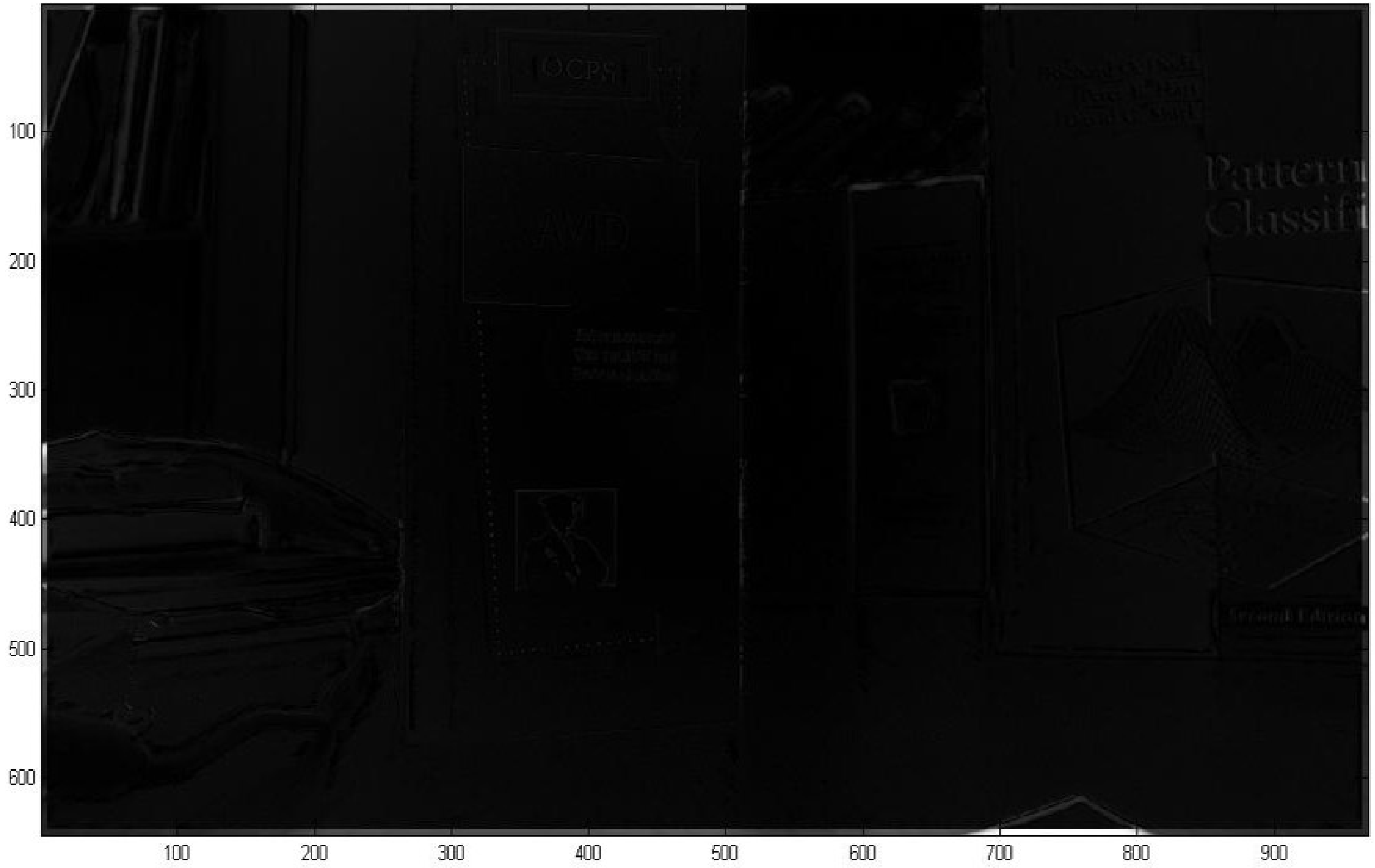
Reconstruction



Shallow DOF Original



Error


100
200
300
400
500
600
100
200
300
400
500
600
700
800
900
Pattern
Classifi

The Problem

- The way I got my results was by looking at a neighborhood around each pixel
- Inspecting such a small scale reduces oblique surfaces to a series of coplanar approximations
- Brute force approach
  - Throw everything at it and see what sticks

Our Goal

- We want to find a way to deal with oblique edges without reducing the problem
- Find a way to determine the appropriate filters needed dynamically
  - Ex: oblique surfaces would need asymmetrical filters because the amount of blur varies across them
- Global optimization with fewer assumptions