Advice to a Young PhD Student

Prof. Vincent Lepetit

- your advisor and you;
- you;
- your research.

- your advisor and you;
- you;
- your research.

Your Advisor and You

you are not a
mere executant

you should not
be on your own

Your Advisor and You

- be independent and take initiative, but
- know when to ask for advice.

Professors are not Gods

- they make mistakes, it is ok to tell them;
- they do not know everything, you also have to read related work;
- but they are experienced, listen to them.

Meet your Advisor Often

Track him/her down if necessary

Meeting your Advisor

How it should **not** go:

- "it does not work"

Meeting your Advisor

How it **should** go:

- "Last time we decided to…"
- "Here is what I did…"
- "It does not work."
- "Here is how I interpret the results…" /
  "what I did to understand the problem",

or ***better***:

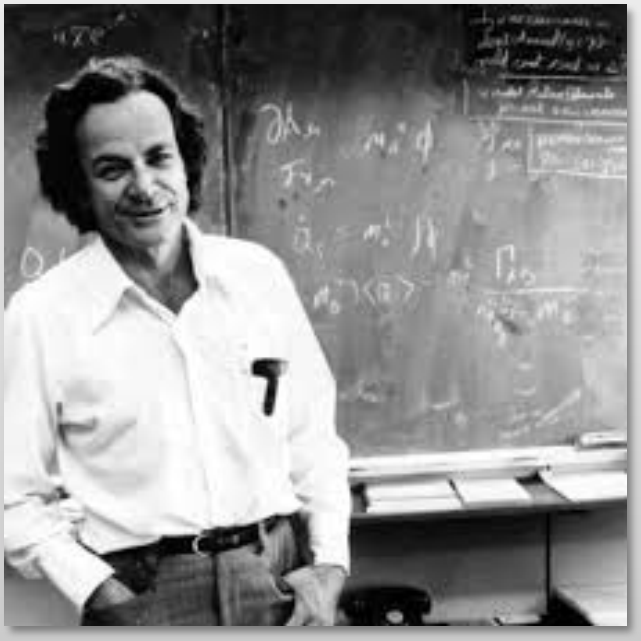
- "Here is what I propose to solve the problem" /
  "what I did to solve the problem"

- if your advisor asks you to do something and you do not understand, **tell him/her!**
- if your advisor asks you to do something you disagree with, **tell him/her (and justify why)**!
- if you realize you made or said a mistake, **tell him/her**!

Doing a PhD can be Tough Sometimes

- your advisor and you;
- you;
- your research.

Work Hard



"I was a normal person who studied hard."
Richard Feyman

Be Efficient

"If it's your job to eat a frog, it's best to do it first thing in the morning.

And if it's your job to eat two frogs, it's best to eat the biggest one first." - Mark Twain

Also check time management books/websites

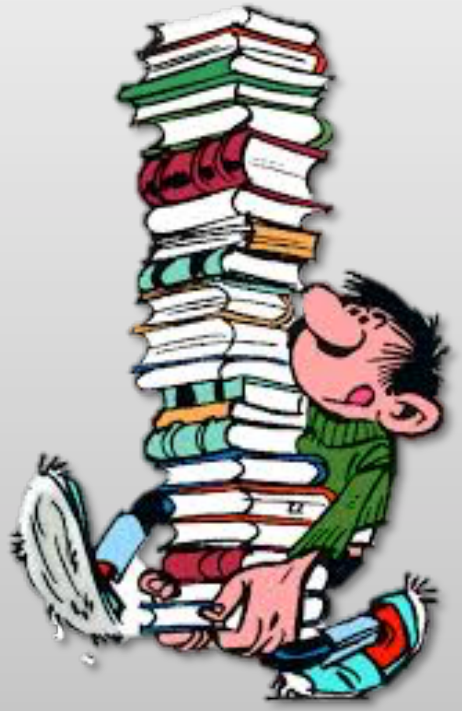
But Take Time for Yourself

[Bill Waterson, Calvin and Hobbes]

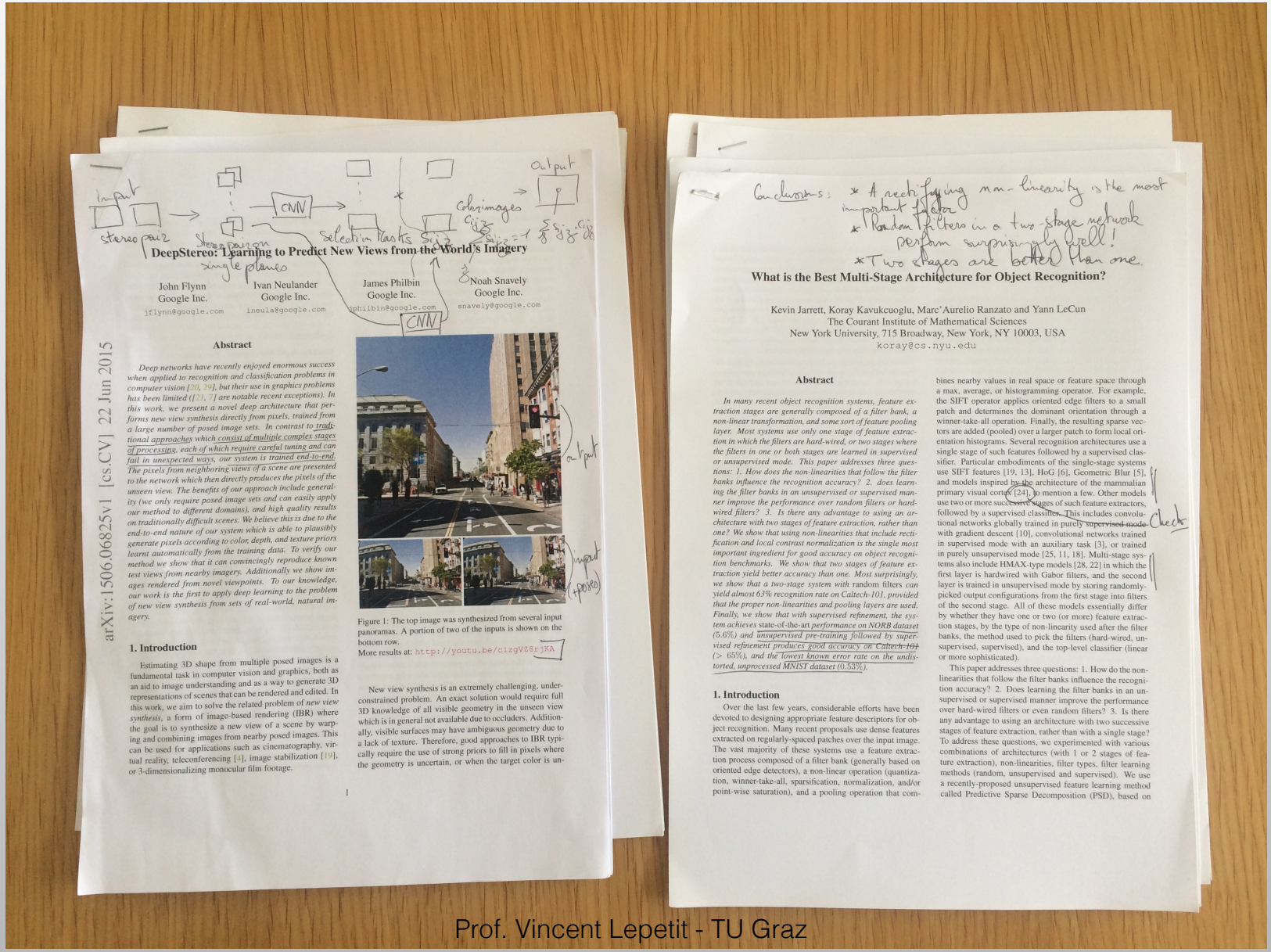
Look for Related Work…

Smart people have probably worked on related problems, or a similar solution.

Focus on the main conferences and journals of the field.



...and Keep Track of It



DeepStereo: Learning to Predict New Views from the World's Imagery

John Flynn, Google Inc. jflynn@google.com
Ivan Neulander, Google Inc. ineula@google.com
James Philbin, Google Inc. jphilbin@google.com
Noah Snavely, Google Inc. snavely@google.com

arXiv:1506.06825v1 [cs.CV] 22 Jun 2015

Abstract

Deep networks have recently enjoyed enormous success when applied to recognition and classification problems in computer vision [20, 29], but their use in graphics problems has been limited ([21, 7] are notable recent exceptions). In this work, we present a novel deep architecture that performs new view synthesis directly from pixels, trained from a large number of posed image sets. In contrast to traditional approaches which consist of multiple complex stages of processing, each of which require careful tuning and can fail in unexpected ways, our system is trained end-to-end. The pixels from neighboring views of a scene are presented to the network which then directly produces the pixels of the unseen view. The benefits of our approach include generality (we only require posed image sets and can easily apply our method to different domains), and high quality results on traditionally difficult scenes. We believe this is due to the end-to-end nature of our system which is able to plausibly generate pixels according to color, depth, and texture priors learnt automatically from the training data. To verify our method we show that it can convincingly reproduce known test views from nearby imagery. Additionally we show images rendered from novel viewpoints. To our knowledge, our work is the first to apply deep learning to the problem of new view synthesis from sets of real-world, natural imagery.

1. Introduction

Estimating 3D shape from multiple posed images is a fundamental task in computer vision and graphics, both as an aid to image understanding and as a way to generate 3D representations of scenes that can be rendered and edited. In this work, we aim to solve the related problem of *new view synthesis*, a form of image-based rendering (IBR) where the goal is to synthesize a new view of a scene by warping and combining images from nearby posed images. This can be used for applications such as cinematography, virtual reality, teleconferencing [4], image stabilization [19], or 3-dimensionalizing monocular film footage.

Figure 1: The top image was synthesized from several input panoramas. A portion of two of the inputs is shown on the bottom row.
More results at: http://youtu.be/cizgVZ8rjKA

New view synthesis is an extremely challenging, under-constrained problem. An exact solution would require full 3D knowledge of all visible geometry in the unseen view which is in general not available due to occluders. Additionally, visible surfaces may have ambiguous geometry due to a lack of texture. Therefore, good approaches to IBR typically require the use of strong priors to fill in pixels where the geometry is uncertain, or when the target color is un-

1

What is the Best Multi-Stage Architecture for Object Recognition?

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato and Yann LeCun
The Courant Institute of Mathematical Sciences
New York University, 715 Broadway, New York, NY 10003, USA
koray@cs.nyu.edu

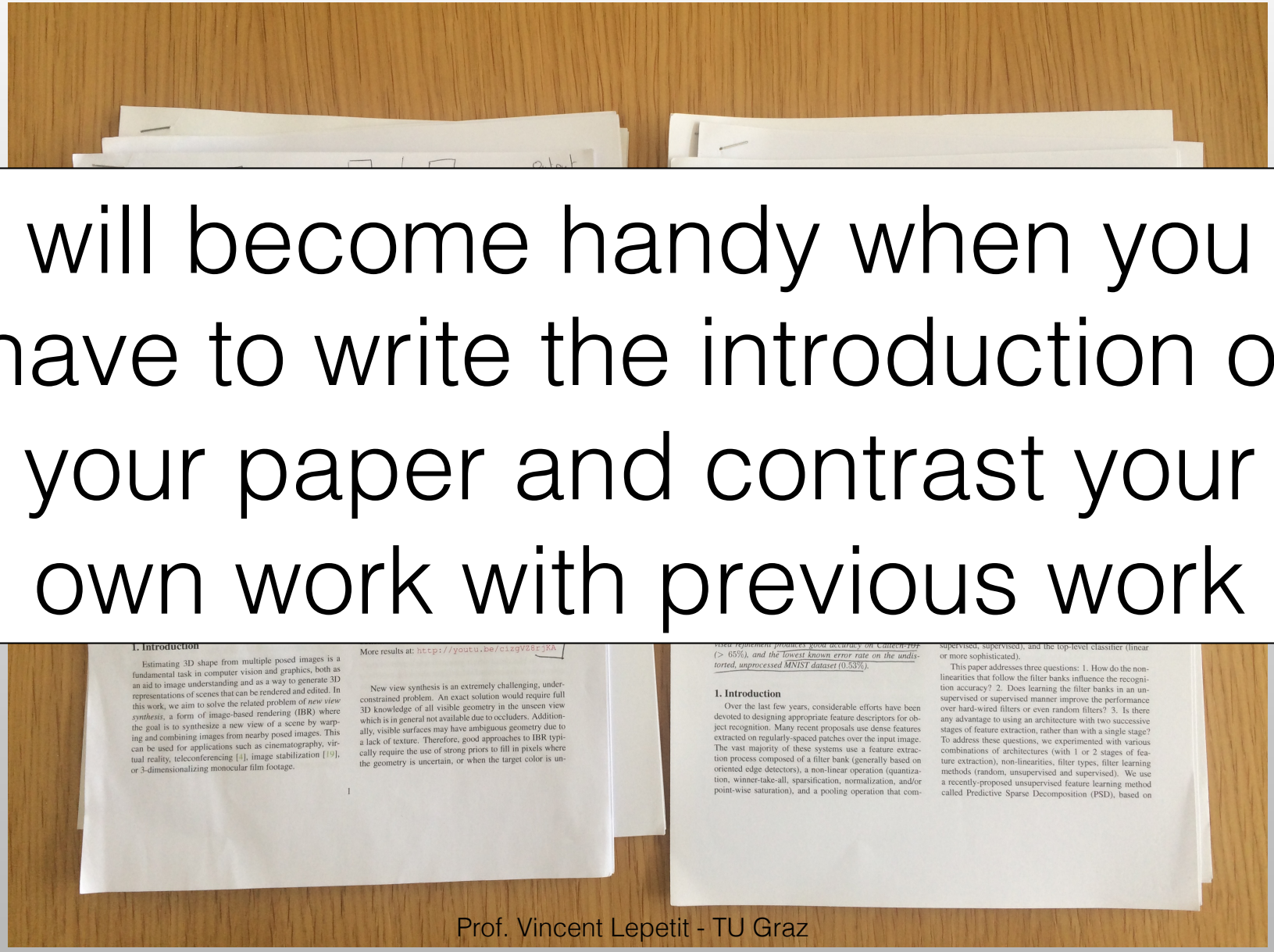
Abstract

In many recent object recognition systems, feature extraction stages are generally composed of a filter bank, a non-linear transformation, and some sort of feature pooling layer. Most systems use only one stage of feature extraction in which the filters are hard-wired, or two stages where the filters in one or both stages are learned in supervised or unsupervised mode. This paper addresses three questions: 1. How does the non-linearities that follow the filter banks influence the recognition accuracy? 2. does learning the filter banks in an unsupervised or supervised manner improve the performance over random filters or hard-wired filters? 3. Is there any advantage to using an architecture with two stages of feature extraction, rather than one? We show that using non-linearities that include rectification and local contrast normalization is the single most important ingredient for good accuracy on object recognition benchmarks. We show that two stages of feature extraction yield better accuracy than one. Most surprisingly, we show that a two-stage system with random filters *can yield almost 63% recognition rate on Caltech-101, provided that the proper non-linearities and pooling layers are used. Finally, we show that with supervised refinement, the system achieves* state-of-the-art *performance on NORB dataset (5.6%) and unsupervised pre-training followed by supervised refinement produces good accuracy on Caltech-101 (> 65%), and the lowest known error rate on the undistorted, unprocessed MNIST dataset* (0.53%).

1. Introduction

Over the last few years, considerable efforts have been devoted to designing appropriate feature descriptors for object recognition. Many recent proposals use dense features extracted on regularly-spaced patches over the input image. The vast majority of these systems use a feature extraction process composed of a filter bank (generally based on oriented edge detectors), a non-linear operation (quantization, winner-take-all, sparsification, normalization, and/or point-wise saturation), and a pooling operation that combines nearby values in real space or feature space through a max, average, or histogramming operator. For example, the SIFT operator applies oriented edge filters to a small patch and determines the dominant orientation through a winner-take-all operation. Finally, the resulting sparse vectors are added (pooled) over a larger patch to form local orientation histograms. Several recognition architectures use a single stage of such features followed by a supervised classifier. Particular embodiments of the single-stage systems use SIFT features [19, 13], HoG [6], Geometric Blur [5], and models inspired by the architecture of the mammalian primary visual cortex [24], to mention a few. Other models use two or more successive stages of such feature extractors, followed by a supervised classifier. This includes convolutional networks globally trained in purely supervised mode with gradient descent [10], convolutional networks trained in supervised mode with an auxiliary task [3], or trained in purely unsupervised mode [25, 11, 18]. Multi-stage systems also include HMAX-type models [28, 22] in which the first layer is hardwired with Gabor filters, and the second layer is trained in unsupervised mode by storing randomly-picked output configurations from the first stage into filters of the second stage. All of these models essentially differ by whether they have one or two (or more) feature extraction stages, by the type of non-linearity used after the filter banks, the method used to pick the filters (hard-wired, unsupervised, supervised), and the top-level classifier (linear or more sophisticated).

This paper addresses three questions: 1. How do the non-linearities that follow the filter banks influence the recognition accuracy? 2. Does learning the filter banks in an unsupervised or supervised manner improve the performance over hard-wired filters or even random filters? 3. Is there any advantage to using an architecture with two successive stages of feature extraction, rather than with a single stage? To address these questions, we experimented with various combinations of architectures (with 1 or 2 stages of feature extraction), non-linearities, filter types, filter learning methods (random, unsupervised and supervised). We use a recently-proposed unsupervised feature learning method called Predictive Sparse Decomposition (PSD), based on



will become handy when you have to write the introduction of your paper and contrast your own work with previous work

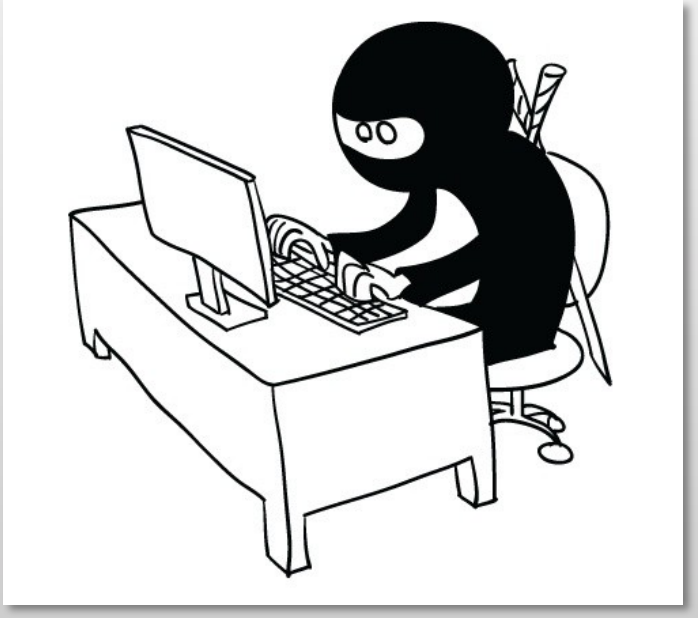
Be an Efficient Coder

Reviewers don't care about the quality of your code.

Coding is for testing an idea, not for a product.

But, if you have something real good *and* if you are prepared to maintain the code, you can make it publicly available (under GPL for instance).

About Coding



Code progressively: Test your ideas and your code on intermediate problems first.

Be efficient, code only what you need.

Save *everything* (code, parameters, results)

- your advisor and you;
- you;
- your research.

The Main Goal of your PhD

1. To develop *good* research;
2. *Communicate* about it so that other people can profit from it

Quality is more important than quantity

Applied Sciences

(Homer:) Look, I just want to know how to invent things.

(Professor Frink:) All you have to do is think of things that people need but which don't exist yet.

identify a problem
about current research

propose a solution

more difficult,
stronger impact

important in practice
(if the method is
actually useful)

improved method

existing method

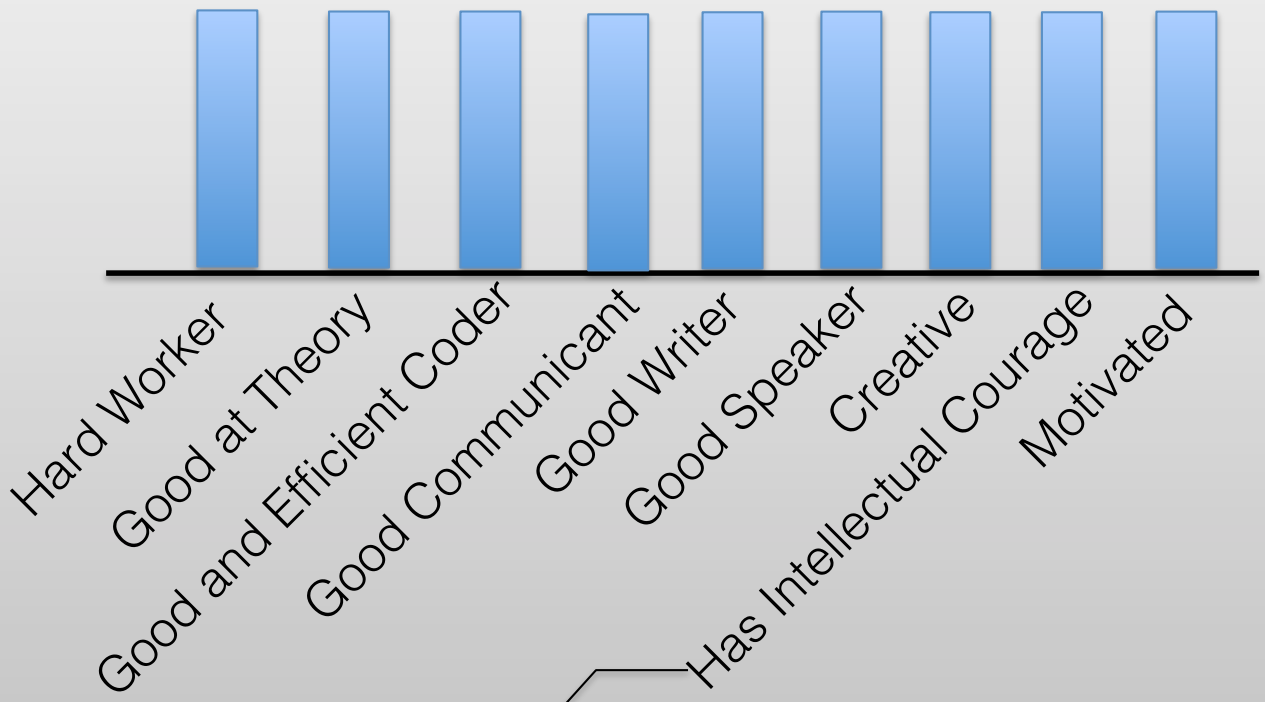
Take Some Time to Read about Science

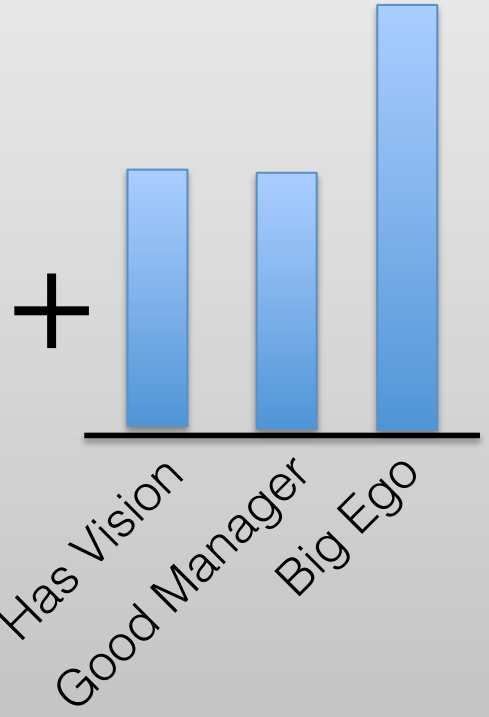
For example:

- *Richard Feynman*;
- The Evolution of *Physics by Albert Einstein and Leopold Infeld*;
- The Selfish Gene *by Richard Dawkins.*

You Need to Learn to Write a Paper and Give a Talk

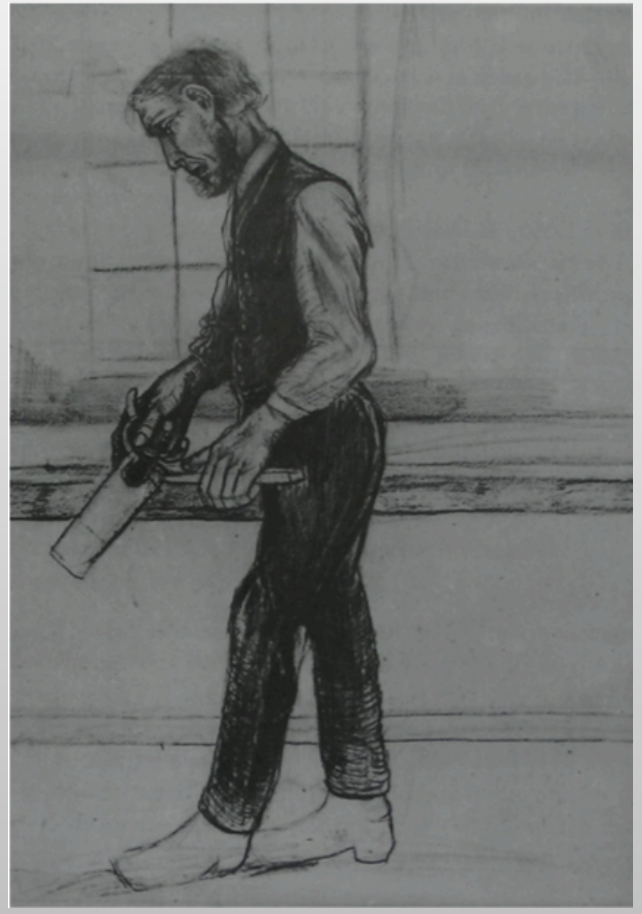
In Summary

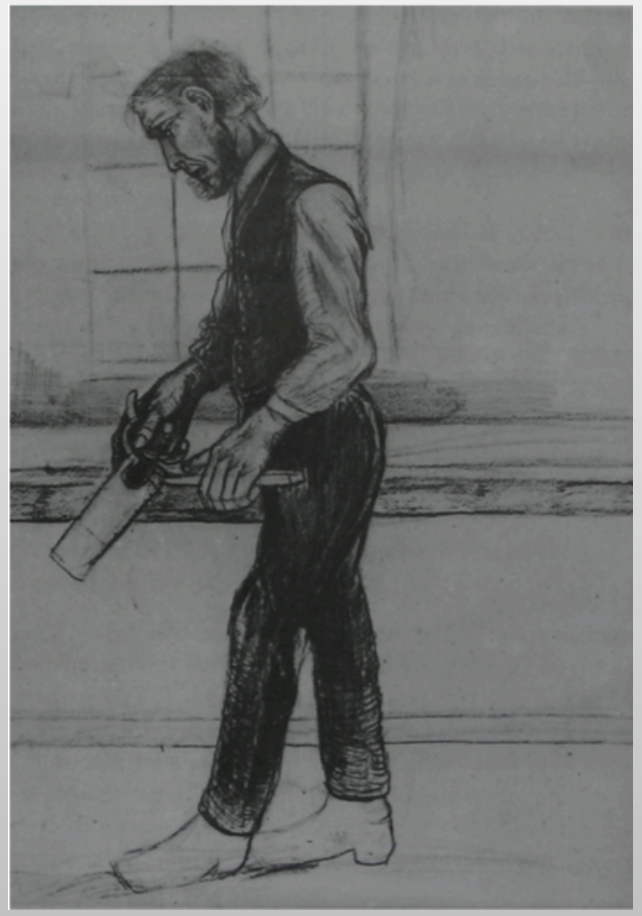


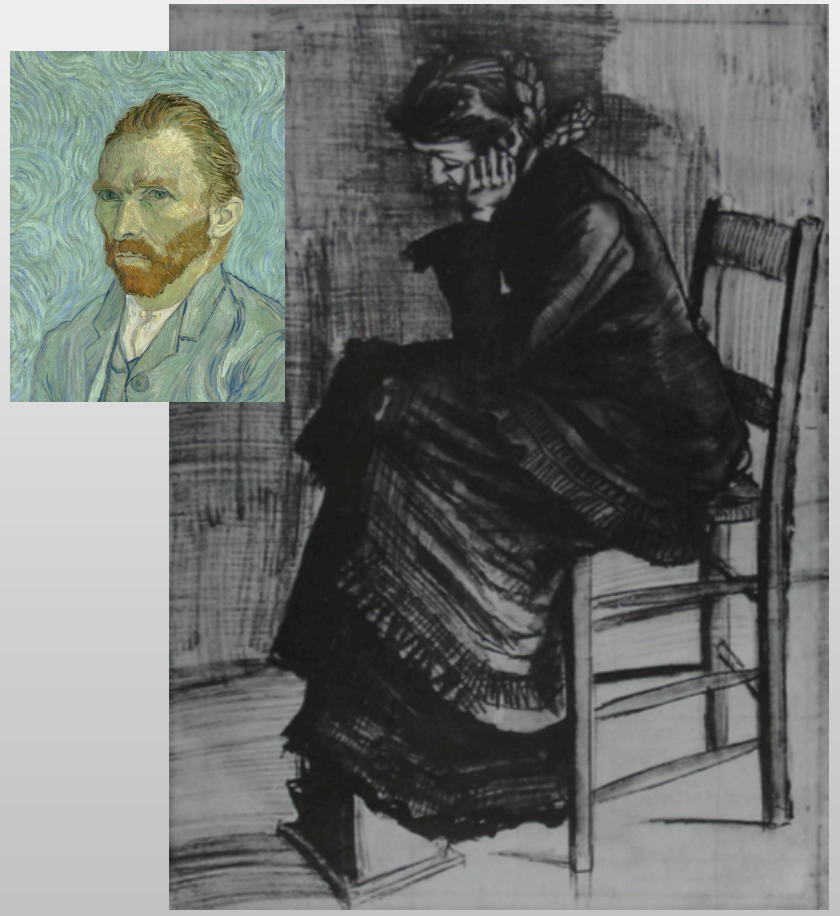


(*Check*: http://www-sop.inria.fr/members/Arnaud.Legout/Documents/YouAndYourResearch.pdf)

for a future academic position







two years later

Next weeks:

- Writing a paper;
- Giving a talk.