Diplomarbeit

An Analysis of Successful Approaches to Human Pose Estimation

Christoph Lassner

07.05.2012
Revised Version (11.06.2012)

Universität Augsburg
Fakultät für Angewandte Informatik
Multimedia Computing and
Computer Vision Lab

Reviewer
Prof. Dr. Rainer Lienhart

Second Reviewer
Prof. Dr. Elisabeth André

Supervisor
M. Sc. Thomas Greif

Abstract

The field of Human Pose Estimation is developing fast and lately leaped forward with the release of the Kinect system. That system reaches a very good performance for pose estimation using 3D scene information, however pose estimation from 2D color images is not solved reliably yet. There is a vast amount of publications trying to reach this aim, but no compilation of important methods and solution strategies. The aim of this thesis is to fill this gap: it gives an introductory overview over important techniques by analyzing four current (2012) publications in detail. They are chosen such, that during their analysis many frequently used techniques for Human Pose Estimation can be explained. The thesis includes two introductory chapters with a definition of Human Pose Estimation and exploration of the main difficulties, as well as a detailed explanation of frequently used methods. A final chapter presents some ideas on how parts of the analyzed approaches can be recombined and shows some open questions that can be tackled in future work. The thesis is therefore a good entry point to the field of Human Pose Estimation and enables the reader to get an impression of the current state-of-the-art.

Kurzbeschreibung

Das Gebiet der automatischen Schätzung der menschlichen Pose in Bilddaten (Human Pose Estimation) entwickelt sich schnell und hat mit der Veröffentlichung der Kinect einen Sprung nach vorn gemacht. Das Kinect-System realisiert Posenschätzung zuverlässig mit Hilfe von 3D Daten, aber eine allgemeine, zufriedenstellende Lösung der Aufgabenstellung auf Grundlage von 2D Farbbildern gibt es bis jetzt noch nicht. Einen Einstieg in dieses aktive Forschungsgebiet zu finden, gestaltet sich schwierig, da zwar viele Forschungsveröffentlichungen aber keine Zusammenstellungen der wichtigsten Lösungsstrategien existieren. Das Ziel dieser Arbeit ist es, diese Lücke zu füllen: Durch die Analyse von vier aktuellen Publikationen gibt sie einen einführenden Überblick über wichtige Methoden für automatische Posenschätzung. Die vier Publikationen sind so gewählt, dass während der Analyse viele wichtige Methoden aus diesem Gebiet erklärt werden können. Die Arbeit beinhaltet zwei einleitende Kapitel, in denen die Aufgabe der automatischen Posenschätzung definiert und die wichtigsten Probleme benannt sowie Grundlagen erklärt werden. In einem abschließenden Kapitel werden einige Ideen aufgezeigt, wie Teile der analysierten Lösungsansätze zu neuen Ansätzen kombiniert werden können, und offene Fragen genannt, die in zukünftigen Arbeiten beantwortet werden können. Diese Arbeit stellt daher einen guten Einstiegspunkt in das Gebiet der automatischen Schätzung der menschlichen Pose dar und ermöglicht dem Leser, sich einen Eindruck vom aktuellen Stand der Forschung zu machen.

Contents

1 Introduction

Human Pose Estimation (HPE) is usually defined as estimating "[...] the configuration of a person's body parts - or 'pose' - in an image" [JE11, p. 1]. With a wide range of applications from human machine interaction over surveillance to diagnostics and man in the focus of interest, it is one of the most fascinating tasks in the field of computer vision. Nevertheless, it remains unsolved in unconstrained settings.

The latest big step towards a reliable solution has been achieved by Microsoft Research and their team with Shotton et al. by finishing the development of the Kinect [Kin10]. It is the first consumer device with capabilities for real-time markerless motion capture (and HPE respectively) in a home-entertainment scenario. The launch of the device was a great overall success, selling eight million units during the first 60 days on sale [Gui11]. It inspired a lot of professional and hobby researchers to develop applications with the new interaction possibilities: the website kinecthacks.com[1] offers 527 applications at the beginning of 2012 and counting. At the same time, a lot of Kinects are used in research labs, e.g. on the TurtleBot[2]. This widespread interest can act as a clue to what a big impact a significant improvement of the solutions for HPE could have.

Consequently, a lot of research effort is made in working on the matter. With better cameras and the availability of more computational power, many new methods for computer vision are explored. The amount of submitted and accepted papers to the "IEEE Conference on Computer Vision and Pattern Recognition" (a major computer vision conference, in short CVPR) can be taken as an indicator for this development, nearly constantly rising since the first conference took place in 1985 (see Figure 1.1).

Being an active field of research, there are no compilations of successful solution strategies and frequently used techniques for Human Pose Estimation yet. The aim of this thesis is, to fill this gap by analyzing four successful approaches. They are chosen such, that many of the most important and most frequently used techniques for HPE can be explained during their analysis. At the same time, the approaches show how the techniques can be combined to form a well-performing system.

In the following chapter, the task of HPE is defined, difficulties are explored and some frequently used techniques are explained. In each of the chapters three to

[1] http://www.kinecthacks.com/
[2] http://www.willowgarage.com/turtlebot

six, a current approach to HPE is described and analyzed. To conclude, in chapter seven some ideas are presented on how some of the explained techniques can be recombined. Furthermore some open questions are specified that can be tackled in future work. With this structure, the thesis is a good entry-point to the field of HPE and enables the reader to get an impression of the state-of-the-art.

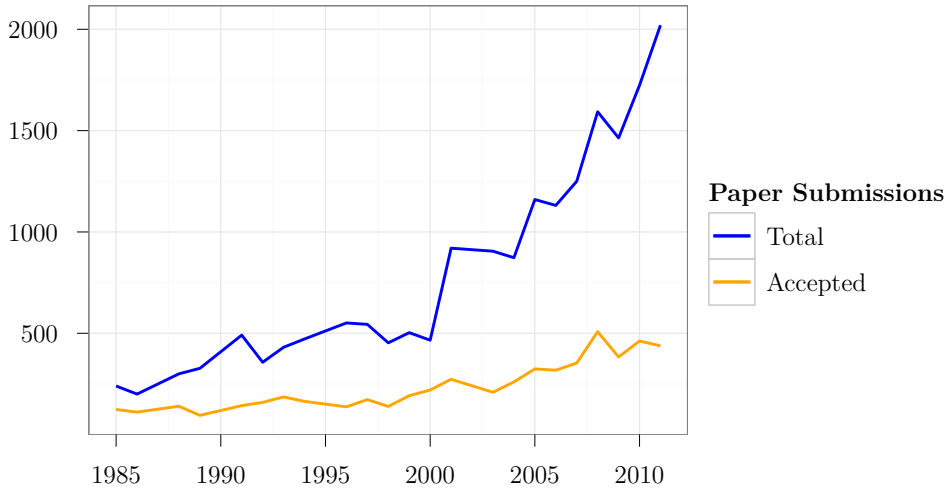


Figure 1.1: Submitted and accepted papers to the CVPR conference plotted by conference year [CVP10, CVP11]. Note that there is no data available for the years 1987, 1990, 1995 and 2002. The values for these years have been interpolated to provide a continuous series over time.

1.1 Applications

Apart from the applications mentioned above, there is a variety of other applications for Human Pose Estimation. Faster computing platforms enable HPE solutions for mobile applications with real-time processing of 3D-Data, e.g. for the Google autonomous cars[3]. This trend might continue and new applications for smartphones and other small devices can become realizable.

At the beginning of 2012, most applications can be categorized in the following three major categories (compare to [MHK06, pp. 1, 2]):

- Surveillance applications:
  - People counting,
  - Crowd flux measurement and congestion analysis,
  - Action and behavior analysis, detection of abnormal activities.

[3] http://googleblog.blogspot.com/2010/10/what-were-driving-at.html

- Control applications:
  - Human machine interaction (e.g. Microsoft Kinect, Playstation Move[4]),
  - Robot control,
  - Virtual reality design.
- Analysis applications:
  - Athlete performance analysis (e.g. for swimmers [Zec11]),
  - Automation (e.g. control of airbags, sleeping detection, pedestrian detection),
  - Orthopedic patient diagnosis (an overview of developments in this specific field can be found in [ZH08]),
  - Video annotation and retrieval,
  - Video compression (e.g. for video conferences).

1.2 Related Work and Related Tasks

Due to the uncommon approach of this thesis, no similarly structured work could be found in the field of HPE. To give an overview of surveys with further references, the most important ones are briefly discussed in the following paragraphs. Many of them focus on different, but closely related tasks to HPE. An overview of related tasks together with a short explanation of the difference to HPE can be found in Figure 1.2.

The surveys by Moeslund et al. [MG01, MHK06] give an overview over publications in the more general field of "Human Motion Capture and Analysis" from 1980 to 2006. They have the aim to cover all papers published on the topic in this timeframe, and structure them using a functional taxonomy as presented in [MG01]. For this task, the steps "Initialization", "Tracking", "Pose estimation" and "Recognition" are used and the papers are classified according to their solution for a specific step.

In 2007, Poppe published a survey [Pop07] focusing on Human Motion Analysis. He describes the difference to human pose estimation as: "When poses are estimated over time, the term human motion analysis is used" [Pop07, p. 1]. As stated in this definition, the task of HPE is a part of the problem of Human Motion Analysis and can be regarded as closely related. This survey does not claim to be complete, but to "summarize the characteristics of and challenges presented by markerless vision-based human motion analysis" [Pop07, p. 1]. Poppe classifies the discussed approaches due to their type of classification algorithm, "model-based,

[4] http://www.playstation.com/psmove/

generative" and "model-free, discriminative". For the model-based approach, the steps "Modeling" and "Estimation" are discussed as sub-steps.

From 2007 on, several further surveys have been published. Most of them concentrate on publications for a specific application of HPE. Zhou and Hu give an overview of developments in the field of Human Motion Tracking for rehabilitation [ZH08]. Wei and Yunxiao focuses on Human Motion Recognition and propose a new taxonomy for that field [WY09]. The field of Pedestrian Detection is covered in a survey of Enzweiler and Gavrila [EG09]. This survey includes extensive experiments with a new proposed dataset for evaluation. In 2010 Poppe published another survey [Pop10], now focusing on Human Action Recognition. In [WRB11], Weinland et al. describe approaches to Action Representation, Segmentation and Recognition. This is to our knowledge the latest survey in the related fields.

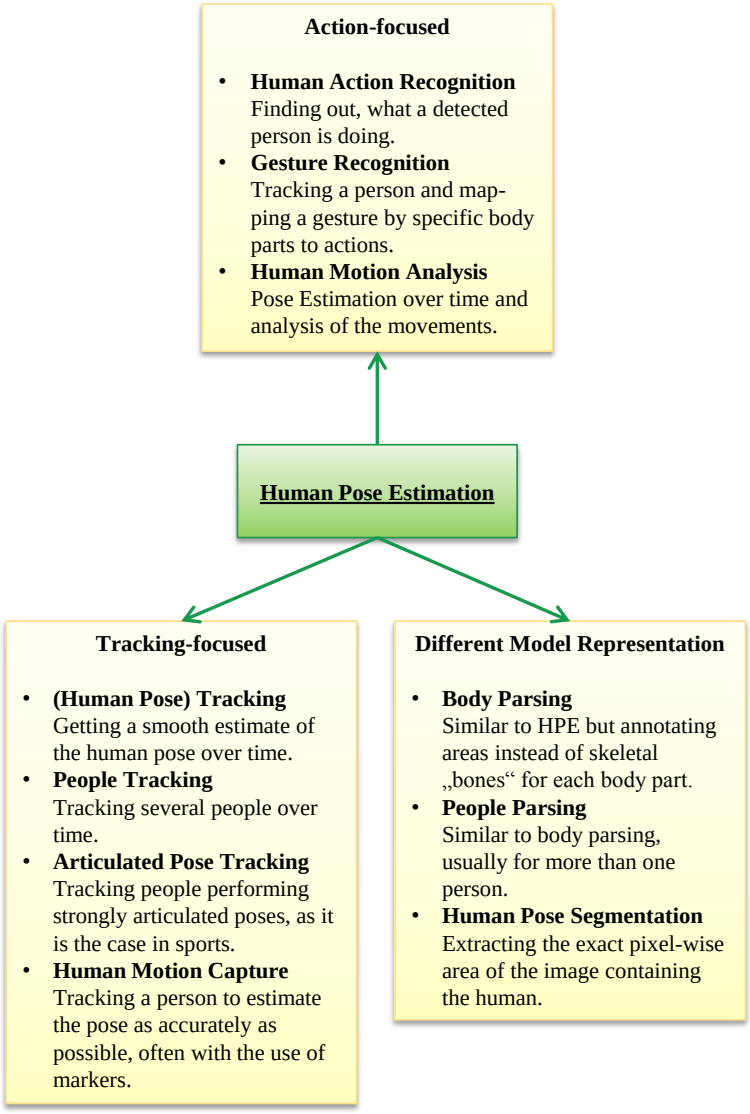


Figure 1.2: Related tasks to Human Pose Estimation.

2 Principles of Human Pose Estimation

2.1 Theoretical Task Analysis

The most general definition of the task of Human Pose Estimation is to estimate the position of the pose-defining human body parts (ankles, knees, hips, shoulders, elbows, hands and head) within digital scene information. As of the date of the publication of this thesis, the most common information providers for HPE are 2D color image sensors and 3D depth image sensors with a resolution of 320×240 pixels up to 1920×1080 pixels.

Figure 2.1 gives an overview of the image formation steps and of the searched function for HPE from sensor information based on light emission from objects. The concept of the "Semantic World" in the top left corner describes the set of all information about the physical world under a certain interpretation. This interpretation can be chosen freely, but plays an important role for the pose estimation step, since it specifies which things are interpreted as body parts and joints.

The bottom part of the figure shows the steps for image formation from the physical world: the world is illuminated, the reflected light is concentrated on a measurement plane and the resulting measurements are sampled to obtain a digital image. These steps unfortunately involve non-injective projection functions, which means that they are not invertible from the obtained image data (see Section 2.1.1). This makes the specification of the searched HPE function (red arrow) very hard. In the following two sections, several of the part problems of the task are explained in detail.

2.1.1 Image Formation

Sensors make use of the perspective projection to collect scene data on their capturing device. In the following paragraph, the Pinhole Imaging Model is explained. It is a general model, specifying the basic point transformation applicable for most imaging devices.

Figure 2.2 shows an illustration of the model. The light from the candle on the right hand travels through a pinhole. It hits the image plane, where a capturing

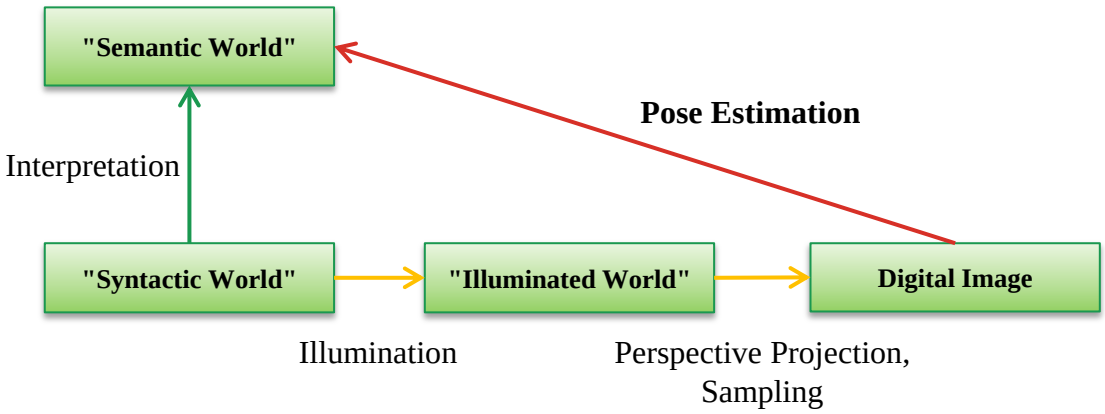


Figure 2.1: Schematic overview of the image formation steps and the pose estimation task. The interpretation can be chosen freely; the transformations marked with yellow arrows are not injective and can not be inverted. The red arrow represents the searched function for the pose estimation task.

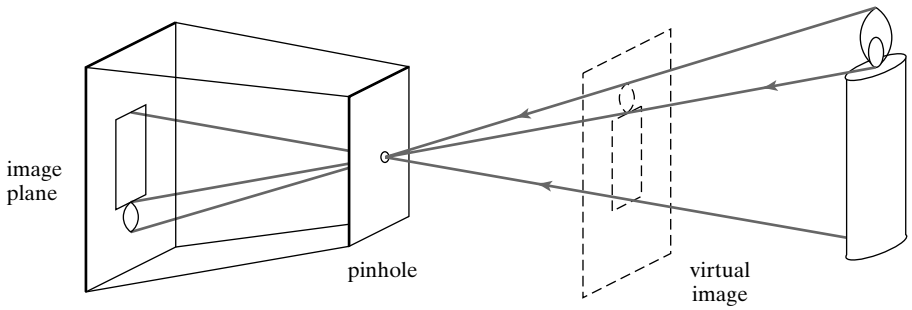


Figure 2.2: The Pinhole Imaging Model [FP02, p. 4].

device is installed. The image is received inverted, because the light rays pass through the pinhole and keep their straight direction. In the theoretical model, only one light ray per object point travels through the hole.

The "virtual image" shown between the pinhole and the candle is a theoretical concept. It is located "outside" of the camera, just as far from the pinhole as the image plane and thus is not inverted.

A camera would have a lens instead of the pinhole and a CCD or CMOS sensor installed at the image plane, but the basic properties of the projection remain the same. The projection function specifying where a point from the outer world is registered on the sensor is called Central Projection. The equations describing it can be derived in few steps.

Mathematical Model Let the origin of the camera coordinate system O coincide with the position of the pinhole and let Π be the image plane (see Figure 2.3). The vector k is defined as a vector perpendicular to the image plane, thus O has the

distance f (the focal distance) to Π along k. With the two orthogonal vectors i and j, the set (i, j, k) forms an orthogonal basis for the camera coordinate system.

The line perpendicular to Π and passing through the pinhole is called the *optical axis*, and the point C where it pierces Π is called the *image center*. "This point can be used as the origin of an image plane coordinate frame, and it plays an important role in camera calibration procedures" [FP02, p. 5].

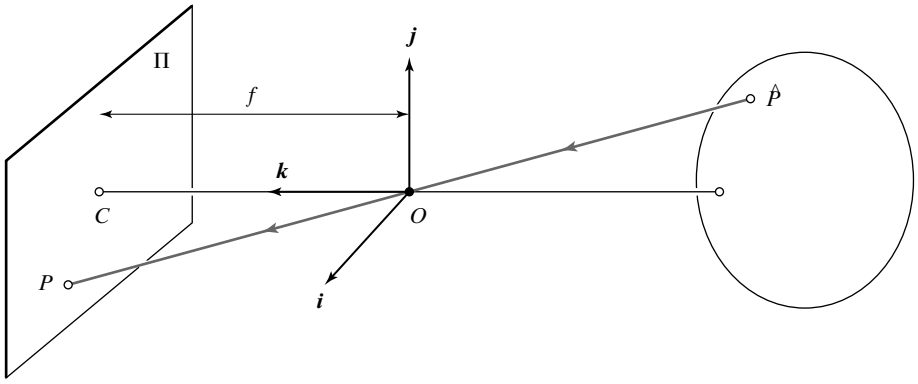


Figure 2.3: Central Projection of point $\hat{P}$ on P (compare to [FP02, p. 6]).

Central Projection Let $\hat{P} = (\hat{x}, \hat{y}, \hat{z})$ be any point in camera coordinates and $P = (x, y, z)$ its image. The points $\hat{P}$, O and P are collinear, since P is created by the (straight) ray of light from $\hat{P}$ through the pinhole in O. Because of this, $\overrightarrow{OP} = \lambda \cdot \overrightarrow{O\hat{P}}$, for a specific λ, so

$$\begin{aligned} x &= \lambda \cdot \hat{x} \\ y &= \lambda \cdot \hat{y} \\ z &= \lambda \cdot \hat{z}. \end{aligned} \tag{2.1}$$

From this follows:

$$\lambda = \frac{x}{\hat{x}} \wedge \lambda = \frac{y}{\hat{y}} \wedge \lambda = \frac{z}{\hat{z}}, \text{ subject to } \hat{x} \neq 0, \hat{y} \neq 0, \hat{z} \neq 0. \tag{2.2}$$

The point P is registered at the image plane at depth $z = f$, thus the values for x and y are

$$\begin{aligned} \frac{x}{\hat{x}} = \frac{f}{\hat{z}} &\Rightarrow x = f \cdot \frac{\hat{x}}{\hat{z}} \\ \frac{y}{\hat{y}} = \frac{f}{\hat{z}} &\Rightarrow y = f \cdot \frac{\hat{y}}{\hat{z}}, \text{ subject to } \hat{z} \neq 0. \end{aligned} \tag{2.3}$$

All sensors apply this transformation to environmental points to make measurements at the image plane. This causes a loss of information, since the function is not injective, thus not bijective and can not be inverted. This theoretical property has severe impact on the difficulty of HPE (and many computer vision problems in general). Pose estimation solely from 2D color images can only estimate the former 3D positions of the objects reflecting the measured light.

Since the projection function is not invertible, these estimations can fail badly. For an example, see Figure 2.4. It shows a "Street Art" faked 3D scene on an appropriately colored flat surface. Though this is an unusual example of art specifically designed to create a depth illusion, it illustrates the difficulty of the problem to infer depth information from scene color in general. These aspects suggest statistical approaches to deal with the infeasible parts of the pose estimation function.

(a) Street Art fake 3D image.

(b) The same scene from the side, unveiling the depth as illusion.

Figure 2.4: Artificial perspective 3D effects from image color [Sta12].

Even with 3D scene information from a local sensor, occlusions still lead to problems. However, pose estimation and background subtraction get easier, and with the Kinect system, a reliable pose estimation system based on 3D scene information is available.

The process of taking a digital image of the scene poses many problems, of which just the two most important were mentioned in this section. Several books exist that discuss the image formation process and also give solutions for some of the occurring problems, e.g. [JÖ5, Sze10].

2.1.2 Variety of Human Appearance and Pose

Independent of the image formation process, the variety of human appearance itself must be handled. With a huge variety of skin colors and their appearance under different lighting conditions, it is hard to use any general color cues to localize

humans. A broad variety in clothing and therefore different body shapes (e.g. shape differences of people in bikinis and ski suits) prohibit the use of strict shape models.

Additionally, the human body can show very different articulations of the extremities. The shoulder joint, the most flexible of human joints, allows to lift the arm about 90° forwards and push it about 45° backwards. At the same time, the arm can be lifted sideways about 90° and turned around its own axis for about 120°. Additional to these movements, the joint can be slightly moved itself and the shoulder blade further enhances the movement possibilities (see [Mar03, p. 6]).

Summing up these findings, it seems that the sheer variety of possible appearance of humans in images can only be captured by statistical approaches with a lot of training data. Clever training strategies can reduce this necessity for data. Depending on the scenario, still a lot of variance must be captured and many possibilities must be evaluated to create a system that can handle pose estimation in general.

2.2 Relevant Fields

Many scientific fields can contribute to solve the aforementioned problems. The following list shows some of them and gives examples of what they can provide to improve HPE performance.

Neuroscience The Merriam-Webster Medical Dictionary defines Neuroscience as "a branch (as neurophysiology) of science that deals with the anatomy, physiology, biochemistry, or molecular biology of nerves and nervous tissue and especially their relation to behavior and learning" [Mer12]. The problem of finding out how consciousness "works" and how animals and humans solve vision tasks successfully is far from being solved. Still, the reverse engineering of brain structures can give inspiration for solving similar technical problems. In this way, Artificial Neural Networks were invented and are now successfully applied to various computer vision problems.

Philosophy The field of philosophy deals with "[...] gaining insight into questions about knowledge, truth, reason, reality, meaning, mind, and value" [Gra99]. This includes questions on how observations can be used to infer information about facts and reality. Knowledge modeling (ontologies), logic modeling and formal logic could play a role in the further development of machine learning.

Physics and Engineering Physics is "the branch of science concerned with the nature and properties of matter and energy. The subject matter of physics includes mechanics, heat, light and other radiation, sound, electricity, magnetism, and the structure of atoms" [Oxf12]. The research and description of optical phenomena is important for computer vision and can contribute to

the development of sensors. New sensor technologies can have a big impact on the development of HPE methods.

Mathematics Mathematics is the foundation of any computation technology. Specifically for the development of HPE, several fields are especially important. Numerical methods must be used to cope with the hardware calculation errors when doing calculations on large amounts of data. Statistical approaches can be used to do classification and find structures in the data. Operations Research methods are used to solve optimization problems efficiently. Graphical Data Analysis is necessary to find visualizations for the usually high-dimensional data and for evaluating methods. Until the date of this publication, there is to our knowledge no satisfactory standard visualization for the evaluation of HPE methods.

Computer Science This field combines methods and technologies from the others and uses them to implement approaches to HPE on computing platforms. Efficient algorithms and data structures are necessary to reach fast computation times. Digital signal processing methods are applied to the raw sensor data to improve data quality. Data mining makes the efficient management and usage of high amounts of data possible.

In the case of Human Pose Estimation, the approach to just imitate "how nature does it" (which has been applied successfully several times in history, e.g. the well-studied and now frequently imitated Lotus effect [Wik12]) can not be applied yet. Scientists are working on reverse engineering the human brain, but with limited success so far (as an example for promising work in progress, see the BlueBrain project [Blu12]).

2.3 Foundations

In this section, some of the basic techniques are introduced that are used by HPE approaches analyzed in this thesis. Not every technique is described in detail, but the explanations in the following paragraphs should give the reader the possibility to grasp the concepts of the analyzed approaches in the following chapters. References to more detailed descriptions are provided for each topic.

2.3.1 Histogram of Oriented Gradients

The Histogram of Oriented Gradients (HOG) is a point descriptor developed by Dalal and Triggs in [DT05]. A point descriptor describes a point by collecting information within a clearly defined environment around it. The HOG descriptor is based on image gradients and uses a rectangular environment to collect information for describing the point. The processing pipeline for the creation of the descriptor is given in Figure 2.5.

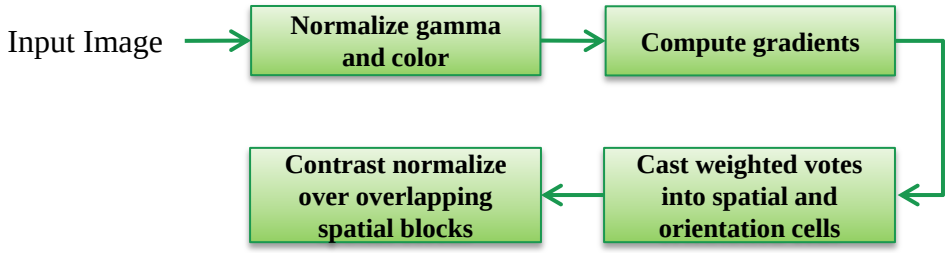


Figure 2.5: HOG descriptor creation pipeline (compare to [DT05, p. 2]).

The HOG descriptor is frequently used for detection and thus calculated for all points in an image. The first step of this procedure is, to normalize the image in gamma and color values, to make possible edges detectable in the next steps.

Then, the gradient image is computed. This can be done efficiently by convolving the image with the two Sobel-filters $[-1\ 0\ 1]$ and $[-1\ 0\ 1]^{\mathrm{T}}$. The results of the two convolutions are converted to one image with gradient magnitudes, and one with gradient orientations.

The next step gives the descriptor its abstraction possibilities. The information is accumulated over several pixels, forming a cell. For human detection, the cell size is chosen as 6×6 pixels. For each pixel within one cell, a vote to an orientation histogram is calculated, equal to its magnitude. For human detection, 9 bins have shown to be the most effective amount of bins. The sign of the orientation is ignored, i.e. the angles are converted to be in $[0; 180]$.

The cells are combined to 3×3 blocks. Several important normalization steps are implemented across blocks and within a cell. For details, please refer to the original paper [DT05]. The result of the method is a grid of point descriptors, i.e. a grid of histograms with nine bins in each cell. Support Vector Machines, which are explained in the following section, can be used for efficient object or human detection on this grid.

2.3.2 Classifiers

Searching for an object or human in digital data can be realized with a detector. It can work directly on the input data or on transformed data, such as an image of point descriptors as described before. The detector can be created using a classifier with at least one class for a “positive” match and one for a “negative” non-match.

Formally, a classifier $\mathcal{C}$ assigns to a sample from the instance space $\mathcal{X}$ a class from all possible classifications $\mathcal{Y}$. The training algorithm building the classifier is called inducer in the following explanations. The inducer $\mathcal{I}$ maps a labeled dataset to a classifier.

In the following two sections, the classifiers used by the analyzed papers are introduced.

2.3.2.1 Decision Trees

Decision Trees are instances of a tree-structured, graphical model. They can classify two or more classes. For a detailed explanation, see [Mit97, pp. 55-80] and [Bis06, pp. 663-666].

To classify an instance, the decision tree is traversed from the root node to a leaf node. The non-leaf nodes are called split nodes. Each split node provides a decision criterion with multiple possible results. The edges leaving the split node each correspond to one of the possible decision outcomes.

Depending on the obtained decision from the currently considered node, the according edge is used to get to the next node for traversal. A leaf node consists of a classification or of information for obtaining a classification, such as a probability distribution. The traversal ends at a leaf node.

For an example, see Figure 2.6. It shows a simple Decision Tree that could be used to determine creditworthiness of people from datasets containing information about income and the duration of employment.

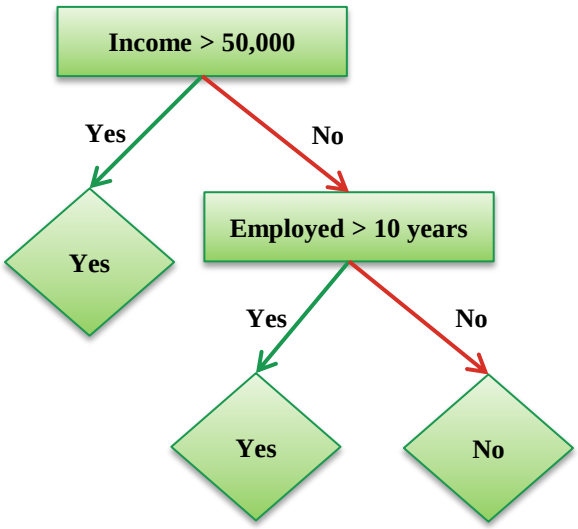


Figure 2.6: Decision tree for samples with the features "Income" and "Employed". It could be used to determine the creditworthiness.

The Shannon Entropy and Information Gain Most decision tree inducers use a measure for "information gain" to find the most informative decision criterions to create a split node. In this section, no decision tree inducer is described, but the information gain is used in Chapter 3 to define one. Thus, this measure is briefly introduced.

The intuition for the use of the information gain measure for the creation of split nodes is, that a split is regarded to be the most informative among a set of possible

splits, if it splits the samples in subsets such that each subset contains samples of as few classes as possible.

This property is measured usually by entropy, which gives an indication for the homogeneity in the set. A frequent choice is the Shannon Entropy. For a set of samples S, it is defined as:

$$\text{Entropy}\,(S) = \sum_{i=1}^{|\mathcal{Y}|} -P_i \cdot \log_2\,(P_i)\,, \tag{2.4}$$

with P_i being the relative frequency of samples with classification i in the set (compare to [Mit97]). With this choice, the entropy is maximal for a set containing equally many samples of each class. As an example, see Figure 2.7. It shows the entropy values for a set containing samples of two classes.

For an illustrative example, consider again the creditworthiness. A set of people being all creditworthy has an entropy of 0, which means it is completely homogeneous. A classifier could simply classify the samples in this set, namely it could give the result "creditworthy". This is the reason, why decision tree inducers try to find splits reducing the entropy in the resulting subsets.

In contrast, consider a set with equally many people receiving a "Yes" and a "No" classification: in this case the entropy equals 1. This mixture of classes cannot be easily classified, and additional splits are necessary to find a resulting classification.

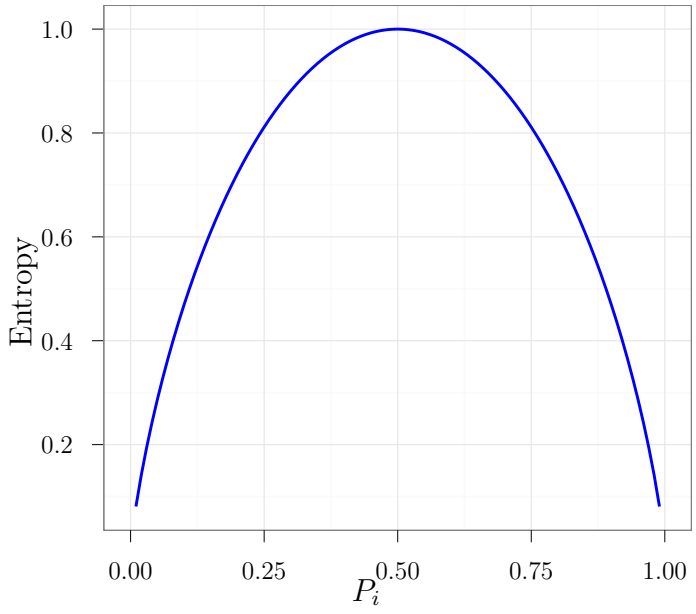


Figure 2.7: Entropy for a set of samples with two classes.

With the definition of entropy, the information gain for a split of a set can be defined. It specifies "[...] the expected reduction in entropy caused by partitioning the examples according to the attribute" [Mit97, p. 57]. The information gain of splitting set S along attribute A is defined as:

$$\text{Gain}\,(S, A) = \text{Entropy}\,(S) - \sum_{v \in \text{values}(A)} \underbrace{\frac{|S_v|}{|S|}}_{\text{weight}} \cdot \underbrace{\text{Entropy}\,(S_v)}_{\text{subset entropy}}. \tag{2.5}$$

It gives the difference between the former entropy and the sum of the entropies in the emerging subsets, weighted with their proportional size.

2.3.2.2 Support Vector Machines

The standard Support Vector Machine (SVM) is a binary linear classifier. It was introduced by Cortes and Vapnik in [CV95]. It is a subtype of the more general class of Kernel Machines. For a detailed explanation including Kernel Machines, see [Bis06].

The SVM is a linear max-margin classifier. "The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points [...]" [Bis06, p. 327]. A linear decision boundary is learned, which has a maximum margin in both directions to the closest data points (see Figure 2.8). Maximizing the margin in both directions gives a unique solution for the decision boundary position.

The decision hyperplane is defined by the equation $y(x) = 0$, with

$$y\,(x) = w^{\mathrm{T}} \phi\,(x) + b, \tag{2.6}$$

where w is an appropriately chosen vector, $\phi\,(x)$ is a fixed feature-space transformation and b an offset. By using appropriate scaling factors, the function values for the closest data points on each side of the decision hyperplane are normalized to 1 and -1 respectively. These closest data points to the decision hyperplane are called support vectors.

The decision of the SVM is determined by applying the linear decision function on the new data point x' and determining the sign of $y\,(x')$. The classification result of a linear SVM can be calculated very efficiently on images by using convolutions.

This standard approach can be generalized with the use of non-linear kernel functions and can also be applied to linearly inseparable data. It is possible to build multiclass SVMs, by "stacking" classifiers, i.e. the first decides whether a sample belongs to class one or not, if not the second decides whether an example belongs to class two or not, etc.

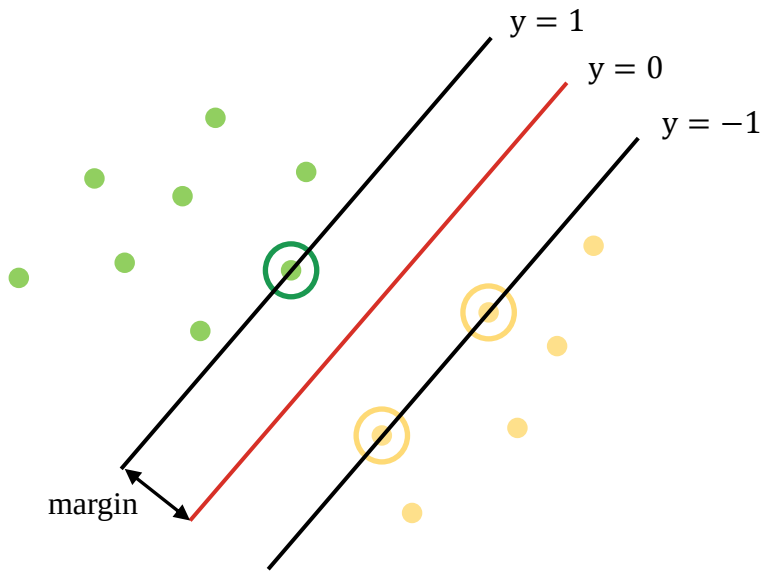


Figure 2.8: Illustration of the decision boundary choice for SVMs (compare to [Bis06, p. 327]). The red line shows the decision boundary, the encircled data points are support vectors. Note, that the position of all other data points is irrelevant for the choice of the decision boundary.

2.3.3 Learning Strategies

Several training strategies exist to enhance the performance of the classifier, maximize the usage of the acquired data and estimate the final performance of the classifier. These techniques are not bound to the use with a specific classifier or inducer.

General Learning Scenario Let $\mathcal{D} = \{v_1, \ldots, v_n\}$ be the set of available annotated data. It consists of n labeled instances $v_i = (x_i \in \mathcal{X}, y_i \in \mathcal{Y})$. A classifier $\mathcal{C}$ maps an unlabeled instance $x \in \mathcal{X}$ to a label $y \in \mathcal{Y}$, and an inducer $\mathcal{I}$ maps a labeled dataset to a classifier $\mathcal{C}$. The notation $\mathcal{I}(\mathcal{D}, x)$ denotes the label assigned to the sample x by the induced classifier from dataset $\mathcal{D}$ and is a short notation for $(\mathcal{I}(\mathcal{D}))(x)$. For all of the following explanations, it is assumed that there exists a distribution on the set of labeled instances and that the dataset consists of independently and identically distributed instances. Note that this is a strong assumption that will be violated by most real world datasets.

The loss function is defined as 0/1 cost function. With $\mathcal{V}$ being the space of correctly labeled instances $\mathcal{V} \subset \mathcal{X} \times \mathcal{Y}$, the accuracy of classifier $\mathcal{C}$ is defined as

$$\text{acc}_{\mathcal{C}} = P(\mathcal{C}(x) = y) \tag{2.7}$$

for a randomly selected instance $(x, y) \in \mathcal{V}$, where the probability distribution to select (x, y) is the same one used to collect the training set.

Given a finite training set, the aim is to estimate the accuracy of a classifier trained with a given inducer on a specific dataset.

2.3.3.1 Holdout

The most intuitive approach to obtain an estimation for the classifier accuracy is to use the inducer only on a part of the dataset (the training set) to train the classifier and to estimate its performance on the remaining samples (the holdout set). The two sets are chosen as mutually exclusive, but exhaustive subsets of $\mathcal{D}$, so formally $\mathcal{D} = \mathcal{D}_t \dot{\cup} \mathcal{D}_h$. The estimated accuracy of the classifier on the holdout set is then given by

$$\mathrm{acc}_h = \frac{1}{|\mathcal{D}_h|} \sum_{(x_i, y_i) \in \mathcal{D}_h} \delta\left(\mathcal{I}\left(\mathcal{D}_t, x_i\right), y_i\right), \tag{2.8}$$

where $\delta(a, b)$ is the Kronecker-delta function, being one if $a = b$ and zero otherwise. It is possible to give a confidence interval for the estimated accuracy (see [Koh95, p. 2]).

However, this method makes inefficient use of the data, since the holdout set is not used for the training of the classifier. To get a better estimate of the classifier accuracy, larger holdout sets are favorable; to reach a better classifier accuracy, smaller holdout sets are favorable.

Additionally, adjusting the parameters of the inducer to improve the performance on the holdout set might lead to overfitting. To avoid this, another validation set is necessary. This further reduces the amount of data available for training (compare to [Bis06, p. 32]).

2.3.3.2 Cross-Validation

A technique avoiding these pitfalls is cross-validation. "The dataset is split into k mutually exclusive subsets (the folds) $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$ of approximately the same size" [Koh95, p. 2]. To get an estimation of classifier accuracy, the inducer is used k times. For trial $t \in \{1, \ldots, k\}$, the dataset $\mathcal{D} \backslash \mathcal{D}_t$ is used for training and $\mathcal{D}_t$ is used for testing. The estimate for the classifier accuracy is calculated as

$$\mathrm{acc}_{\mathrm{CV}} = \frac{1}{n} \sum_{t=1}^{k} \sum_{(x_i, y_i) \in \mathcal{D}_t} \delta\left(\mathcal{I}\left(\mathcal{D} \backslash \mathcal{D}_t, x_i\right), y_i\right). \tag{2.9}$$

The estimation of classifier accuracy makes use of the entire dataset, since every sample is once in the test set. However, it is subject to the choice of the k folds: for a very unlucky choice of the folds, each fold could contain samples from which

the inducer cannot construct a capable learner, but if the inducer could make use of the entire dataset it could succeed.

Complete cross-validation tries to avoid this risk, by estimating the classifier accuracy as the average of the $\binom{n}{\frac{n}{k}}$ possible k-fold cross-validation results. However, this very elaborate solution requires a lot of computational power.

An always complete application strategy for cross-validation is the leave-one-out method (n-fold cross-validation). It can be applied when data is particularly sparse.

For a sufficiently high amount of data, a confidence interval can be computed for cross-validation, similar to the holdout method (see [Koh95, p. 3]).

2.3.3.3 AdaBoost

AdaBoost (short for "Adaptive Boosting") is a meta learning strategy for binary classifiers and was developed by Freund and Schapire in [FS97]. It combines the decisions of many weak classifiers to one, more capable classifier.

In an iterative process, the classifiers are combined using the information of a weighted error function. In each iteration, the most capable of the classifiers is added to the ensemble with a specific weight for its vote. The weights for the error function are updated such, that samples more likely to be misclassified by the ensemble so far receive higher weights, whereas samples likely to be classified correctly receive lower weights. Then the next iteration is done. The ensemble classifier output is a majority vote. For a detailed explanation of the process, see [FS97].

2.3.4 Pictorial Structures

The concept of Pictorial Structures was developed originally by Fischler and Elschlager in 1973 in [FE73]. The key idea is to explicitly model the appearance of an object as configuration of specific object parts (see Figure 2.9). Felzenszwalb and Huttenlocher made the approach popular for object detection in 2000 by specifying an efficient detection approach in [FH00]. Felzenszwalb continued to develop object detection algorithms with it [FH05, FGMR10]. It is used for many tasks in computer vision and has also been adopted for Human Pose Estimation (e.g. by Andriluka et al. [ARS09]).

2.3.4.1 Motivation and Modeling Framework

Figure 2.9 illustrates the key concept: deformable objects with very different shapes can be modeled as collection of static, reoccurring object parts. Detectors can be used to detect these parts and their results can be combined. Thus, the detectors

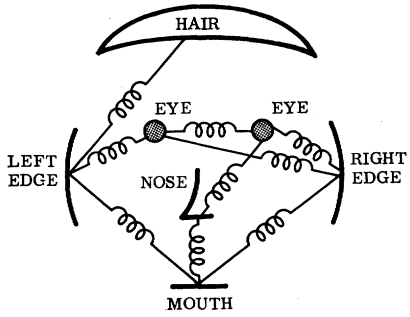


Figure 2.9: Face model by Fischler and Elschlager [FE73, p. 11]. The appearance of the face is modeled as configuration of the face parts. The springs illustrate the connection functions for the parts.

do not have to capture the high variability of the possible locations of the parts within the complex model. Since the human body can show so many different configurations of the body parts, especially the extremities, this technique can help to reduce the variance that must be captured by the detectors.

In a modeling step, the object is modeled as a collection of parts with connections between some of the parts. For the connected parts, a connection function is specified which captures the possible combinations of the object parts. The connection function can be chosen dependent on the object to detect. With this design Pictorial Structures are very flexible and allow the specification of many object types.

Formally, the object model is described with a graph (V, E), consisting of n vertices $V = \{v_1, \ldots, v_n\}$ and edges E. Each vertex represents one object part and the edges capture the connection function between two connected parts. In order to apply the efficient matching algorithms developed by Felzenszwalb and Huttenlocher, it is necessary that the model has a tree structure.

For each object part node v_i, there is an observation node l_i, which represents the current observation (usually location, scale and orientation) of the part. The observations l_i are referred to as part configurations. A complete instance of the detected object is given by the object configuration $L = (l_1, \ldots, l_n)$. For an example model, see Figure 2.10. The depicted model is a standard model for human detection and will be explained in one of the following sections.

2.3.4.2 The Bayesian Matching Framework

After having defined a modeling framework, it is critical to find a matching framework, so that modeled objects can be detected automatically. The matching framework is derived with a Bayesian approach.

From a probabilistic point of view, the best (most probable) configuration L^* for

the model (V, E) in image I with the learned model parameters Θ is searched, i.e.

$$L^* = \arg\max_L P(L|I, \Theta). \tag{2.10}$$

The model parameters consist of three components: $\Theta = (u, E, c)$. The first component, u, contains appearance parameters for the body parts, E are the edges of the graph and indicate which parts are connected, and $c = \{c_{ij} | (v_i, v_j) \in E\}$ are the connection parameters for each connection. All parameters can be learned from data, however E is often fixed in the modeling step (as it has been done for the human model in Figure 2.10). The probability for a configuration is reformulated with the Bayes formula as

$$\begin{aligned} P(L|I, \Theta) &= \frac{P(I|L, \Theta) \cdot P(L|\Theta)}{P(I|\Theta)} \\ &\propto \underbrace{P(I|L, \Theta)}_{\text{Configuration matching prob.}} \cdot \underbrace{P(L|\Theta)}_{\text{Configuration prior prob.}}. \end{aligned} \tag{2.11}$$

The term $\mathrm{P}(I|\Theta)$ describes the probability of observing an image, given the model parameters. This probability can hardly be estimated and is assumed to be equal for all images. Thus, it can be reduced to a constant scaling factor and omitted, since a positive constant factor leaves the order unchanged in the maximization step.

Matching Probability In the simplified representation in Equation 2.11, the matching probability $P(I|L, \Theta)$ occurs, specifying the probability of observing an image, given a configuration and the model parameters. This probability is modeled as the product of the individual likelihoods of observing the object parts at their specified locations:

$$P(I|L, \Theta) = P(I|L, u) = \prod_{i=1}^{n} P(I|l_i, u_i). \tag{2.12}$$

Θ can be reduced to the relevant appearance parameters. The values of the appearance terms $P(I|l_i, u_i)$ are estimated by applying a detector to the image region defined by l_i with parameter u_i. Here the meaning of the appearance parameters becomes clear: they capture the learned parameters for the detector.

The second term $P(L|\Theta)$ in Equation 2.11 specifies the probability for a specific configuration given the model parameters. Note that this probability does not depend on the image. The further modeling steps for $P(L|\Theta)$ are explained in the following section.

2.3.4.3 Derivation of the Prior Probability Representation

The graph describing the Pictorial Structure Model (PSM) must be tree-structured (in the following explanations, the example in Figure 2.10 is used). It represents a graphical probabilistic model, with an edge from v_i to v_j indicating dependency of v_j only on v_i. This includes the assumption, that the probability for a configuration of one part is independent of all other part configurations given the configuration of the parent part. This is a strong assumption which is violated in many cases, e.g. by mutual occlusions of body parts.

The distribution describing the probability for an object configuration is given by

$$P\left(L|\Theta\right) = P\left(l_{root}|\Theta\right) \cdot \prod_{(v_i,v_j)\in E} P\left(l_j|l_i,\Theta\right). \tag{2.13}$$

This also reflects the natural interpretation of the model: the combined probability is the probability for the root part configuration, iteratively multiplied with the probabilities of the child part configurations. For example, the probability for a configuration of the human-model in Figure 2.10 is

$$P\left(L|\Theta\right) = \underbrace{P\left(l_t|\Theta\right)}_{\text{Torso conf.}} \cdot \underbrace{P\left(l_h|l_t,\Theta\right)}_{\text{Head c. given torso c.}} \cdot \underbrace{P\left(l_{rua}|l_t,\Theta\right) \cdot P\left(l_{rla}|l_t,\Theta\right)}_{\text{Right arm c. given torso c.}} \cdot \ldots. \tag{2.14}$$

The conditional probabilities in the part terms can be rewritten as

$$P\left(l_j|l_i,\Theta\right) = \frac{P\left(l_i,l_j|\Theta\right)}{P\left(l_i|\Theta\right)}, \tag{2.15}$$

which leads to the formulation of the entire probability as

$$P\left(L|\Theta\right) = P\left(l_{root}|\Theta\right) \cdot \prod_{(v_i,v_j)\in E} \frac{P\left(l_i,l_j|\Theta\right)}{P\left(l_i|\Theta\right)}. \tag{2.16}$$

Each term $P\left(l_i|\Theta\right)$ in the denominator in this equation occurs as often as an edge leaves the node v_i. The degree $\deg\left(v_i\right)$ of node v_i is defined as the amount of edges coming to or leaving v_i in the graphical model. Since the model is a tree, the amount of edges leaving the node v_i, is exactly $\deg\left(v_i\right) - 1$. This is, because in a tree every node has exactly one parent, except for the root node that has only leaving edges (the amount of edges leaving the root node is $\deg\left(v_{root}\right)$). Thus, the probability can be written as

$$P\left(L|\Theta\right) = P\left(l_{root}|\Theta\right) \cdot \frac{\prod_{(v_i,v_j)\in E} P\left(l_i,l_j|\Theta\right)}{P\left(l_{root}|\Theta\right) \cdot \prod_{v_i\in V} P\left(l_i|\Theta\right)^{\deg(v_i)-1}}. \tag{2.17}$$

This leads to the representation frequently used in literature (e.g. in the derivation of Felzenszwalb and Huttenlocher [FH05, p. 10]):

$$P(L|\Theta) = \frac{\prod_{(v_i,v_j)\in E} P(l_i, l_j|\Theta)}{\prod_{v_i \in V} P(l_i|\Theta)^{\deg(v_i)-1}}. \tag{2.18}$$

A strong approximation is used by assuming that $P(l_i|\Theta) = 1$. This means, that the absolute configuration of object parts is not used for inferring the most likely object configuration. The prior is then formulated as

$$P(L|\Theta) = P(L|E, c) = \prod_{(v_i,v_j)\in E} P(l_i, l_j|c_{ij}). \tag{2.19}$$

The terms $P(l_i, l_j|c_{ij})$ are the prior terms for each pair of connected body parts. These probabilities are estimated using the aforementioned connection function.

Connection Function There are many possible choices for the connection function, dependent on the modeled object. However, it must be possible to express it in a specific form to apply the efficient matching algorithms of Felzenszwalb and Huttenlocher. This form is a Diagonal Gaussian Normal Distribution over the displacement in a common coordinate system, i.e.

$$P(l_i, l_j|c_{ij}) = \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}), \tag{2.20}$$

where additional parameters for the transformation functions T_{ij} and T_{ji} as well as the standard deviations D_{ij} are encoded by the connection parameter c_{ij}. D_{ij} is a diagonal covariance matrix and its entries are learned from the data. For each part, a "connection point" is defined in the parts own coordinate system (for an example, see Figure 2.11). T_{ij} and T_{ji} are bijective functions, determining the connection point position for the connected parts in a common coordinate system, usually the image coordinate system.

The transformation functions are determined in a modeling step and must be chosen such that for an optimal pair of locations, the difference $T_{ij}(l_i) - T_{ji}(l_j)$ becomes as small as possible. Example choices can be found in [FH05, p. 22] and [FH05, p. 29-31]. A possible choice for the use for Human Pose Estimation is explained in Section 2.3.4.5.

2.3.4.4 The Final Optimization Problem

With the results from Equation 2.12 and Equation 2.19, the original Bayesian formulation in Equation 2.11 can be rewritten as

$$\begin{aligned} P(L|I, \Theta) &\propto P(I|L, \Theta) \cdot P(L|\Theta) \\ &= \prod_{i=1}^{n} P(I|l_i, u_i) \cdot \prod_{(v_i,v_j)\in E} P(l_i, l_j|c_{ij}). \end{aligned} \tag{2.21}$$

To simplify the optimization problem, it is possible to take the negative logarithm of this formula. This inverts the problem (now minimization instead of maximization), but keeps the order of ratings for the configurations. Since only the configuration with the maximum probability is searched but not the exact probability value, this loss of information is acceptable.

Defining the matching function $m_i(l_i) = -\log P(I|l_i, u_i)$ and the distance function $d_{ij}(l_i, l_j) = -\log P(l_i, l_j|c_{ij})$ leads to the final minimization problem to find the best matching configuration L^*:

$$L^* = \arg\min_L \left(\sum_{i=1}^{n} m_i(l_i) + \sum_{(v_i,v_j)\in E} d_{ij}(l_i, l_j) \right). \tag{2.22}$$

This minimization problem can be solved very efficiently with generalized distance transforms (see [FH00]).

2.3.4.5 Pictorial Structure Models for Human Pose Estimation

The most commonly used Pictorial Structure Model for Human Pose Estimation models the human body as collection of ten body parts: head, upper and lower arms, torso and upper and lower legs (see Figure 2.10). The edges in the model are usually fixed and not learned from data. The observation nodes represent the observed configuration for a specific body part.

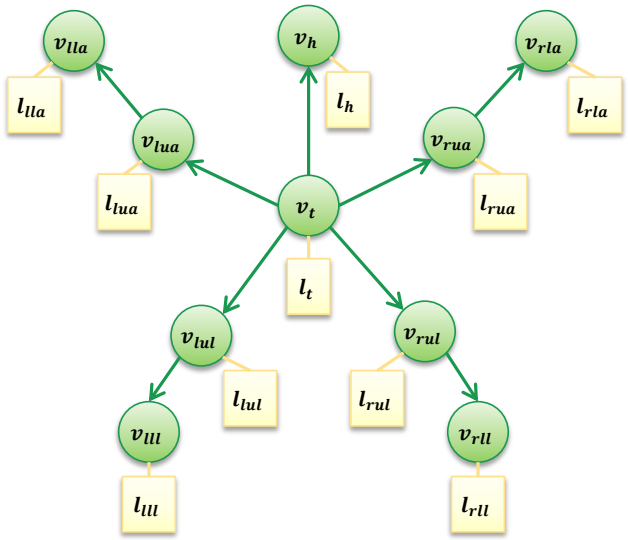


Figure 2.10: Pictorial Structure Model graph for human appearance. Associated observation nodes are shown as yellow rectangles.

The second important modeling step is the choice of the transformation functions T_{ij} and T_{ji}. This choice defines the likeliness of a part combination and thus has a

strong impact on the model matches. There are many possibilities for the definition of these functions. As an example, the definition by Felzenszwalb and Huttenlocher in [FH05] is introduced here.

With the high flexibility of the human body, it is necessary to model the joints between the parts deformable. Still, preferably many constraints should be added, to benefit from additional information in the inference step.

(a) Each part in its own coordinate system (indicated with the crosses) and the position of the connection point (circle).

(b) The ideal configuration of the part pair. This configuration has maximal probability. Difference in any dimension (position, scale, rotation) will result in lower probability values.

Figure 2.11: Two connected parts of a Pictorial Structure Model [FH05, p. 29].

For an illustration of the modeling idea, refer to Figure 2.11. For each of two connected parts, a location for the connection point (x_{ij}, y_{ij}) and (x_{ji}, y_{ji}) is specified in the coordinate system for the according part. In an ideal configuration, both connection points coincide. Additionally, the relative rotation of the parts and similarity of their scale is considered for calculating a final probability. This model setup allows fine tuning in several parameters, while still allowing a lot of variance if needed.

Let $l_i = (x_i, y_i, s_i, \theta_i)$ and $l_j = (x_j, y_j, s_j, \theta_j)$ be the configurations of two connected parts (consisting of x and y coordinate, scale and orientation), and θ_{ij} be the ideal difference between their orientations. Then the probability for the two locations is modeled as product of the Normal Distributions around the optimal configuration in the respective dimension, i.e.

$$\begin{aligned} P\left(l_i, l_j | c_{ij}\right) \quad = \quad & \mathcal{N}\left(x_i' - x_j', 0, \sigma_x^2\right) \cdot \quad (\leftarrow \text{Horizontal diff. of conn. points}) \\ & \mathcal{N}\left(y_i' - y_j', 0, \sigma_y^2\right) \cdot \quad (\leftarrow \text{Vertical diff. of conn. points}) \\ & \mathcal{N}\left(s_i - s_j, 0, \sigma_s^2\right) \cdot \quad (\leftarrow \text{Scale difference}) \\ & \mathcal{M}\left(\theta_i - \theta_j, \theta_{ij}, k\right) \quad (\leftarrow \text{Rotation difference}), \end{aligned} \tag{2.23}$$

where $\mathcal{M}$ is the von Mises distribution (a concept similar to the Normal Distribution in circular coordinate systems), and (x_i', y_i') and $\left(x_j', y_j'\right)$ are the connection point

positions in image coordinates. These coordinates can be obtained by applying the transformations of rotation (R_θ is the matrix for the rotation by angle θ around the origin), scaling, and translation to the part detection points, i.e.:

$$\begin{aligned} \begin{pmatrix} x'_i \\ y'_i \end{pmatrix} &= \begin{pmatrix} x_i \\ y_i \end{pmatrix} + s_i \cdot R_{\theta_i} \begin{pmatrix} x_{ij} \\ y_{ij} \end{pmatrix}, \\ \begin{pmatrix} x'_j \\ y'_j \end{pmatrix} &= \begin{pmatrix} x_j \\ y_j \end{pmatrix} + s_j \cdot R_{\theta_j} \begin{pmatrix} x_{ji} \\ y_{ji} \end{pmatrix}. \end{aligned} \tag{2.24}$$

However, it is necessary to specify $P(l_i, l_j | c_{ij})$ as multidimensional Diagonal Normal Distribution to apply the efficient matching algorithms. This can be achieved by transforming the von Mises Distribution to a Normal Distribution (for the steps, refer to [FH05, p. 30]), resulting in the following definition:

$$\begin{aligned} T_{ij}(l_i) &= (x'_i, y'_i, s_i, \cos(\theta_i + \theta_{ij}), \sin(\theta_i + \theta_{ij})), \\ T_{ji}(l_j) &= \left(x'_j, y'_j, s_j, \cos(\theta_j), \sin(\theta_j)\right), \\ D_{ij} &= \operatorname{diag}\left(\sigma_x^2, \sigma_y^2, \sigma_s^2, \frac{1}{k}, \frac{1}{k}\right), \\ P(l_i, l_j | c_{ij}) &\propto \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}). \end{aligned} \tag{2.25}$$

Note, that the x, y and s components remain unchanged from Equation 2.23 and are just split between the T_{ij} and T_{ji} functions. The remaining two dimensions originate from the conversion of the von Mises distribution. The conversion is also responsible for the "proportional to" sign, instead of an equal sign, since an approximation step is used.

The remaining parameters for each connection are

$$c_{ij} = \left(x_{ij}, y_{ij}, x_{ji}, y_{ji}, \sigma_x^2, \sigma_y^2, \sigma_s^2, \theta_{ij}, k\right), \tag{2.26}$$

and can be learned from training data using maximum likelihood methods (see [FH05, p. 30]).

2.4 Task Definition and Choice of Analyzed Approaches

The aim of this thesis is to give an overview of techniques applied for Human Pose Estimation for 2D color images. The pose is estimated from depictions of humans; the color images originate from any common color sensor camera and contain the human figure at a size of approximately 160 pixels. The images must show sufficient illumination and contrast to show the outlines of the human figure.

Similar restrictions are applied by current publications. Four are chosen to reflect the state of the art and to show most techniques that are applied to realize HPE.

[SFC+11], by Shotton et al. The first paper analyzed describes the algorithms of the Kinect pose estimation system. It is chosen despite the approach is based on 3D scene information, since the Kinect system is the only currently available system to our knowledge that reaches satisfactory pose estimation performance in real-world scenarios. Additionally, the Kinect system might enable for easier data acquisition for new pose estimation approaches. The paper received a best-paper award at the CVPR 2011.

[SKN10], by Singh et al. Contextual knowledge, especially about human interaction, can significantly improve pose estimation performance. This paper describes an approach incorporating contextual knowledge into the pose estimation process. It is based on a combination of Pictorial Structures and a pose estimation approach by Deva Ramanan [Ram07]. The work by Singh et al. received a best paper award at the Conference on Structured Models in Computer Vision 2010.

[JE10], by Johnson and Everingham The requirement of large datasets has been explained already in this introductory chapter. Johnson and Everingham propose a new, large scale data acquisition method for HPE. To meet the requirements of the large datasets, they introduce Pictorial Structures in an extended framework called Clustered Pictorial Structure Models. It can deal with extremely articulated poses and reaches very high pose estimation performance on the major HPE evaluation dataset.

[WK11], by Wang and Koller This publication by Wang and Koller explains how high level information from Pictorial Structure matching can be combined with low level per pixel segmentation information. An energy function is assembled that combines these different levels of information. Wang and Koller apply relaxed dual composition to include infeasible energy functions in the optimization process. Their concept combines multiple methods and improves their common results.

3 Real-time Pose Recognition in Parts from Single Depth Images

The Microsoft Kinect [Kin10] is a game controller released at the end of 2010. It made a new interaction paradigm possible: with real-time motion capture, the device allows to control games and applications directly by body movements without physical contact to any device. It works reliably in home-entertainment environments.

To achieve this performance, it relies on depth information from the Kinect sensor. The algorithms used to realize real-time HPE are in parts explained in the publication "Real-time Pose Recognition in Parts from Single Depth Images" by the developers Shotton et al. [SFC+11]. In this chapter the approach by Shotton et al. is analyzed, complemented with a short explanation of the depth estimation process of the Kinect sensor.

3.1 Target Scenario and Processing Steps

At the beginning of 2012, two versions of the Kinect system are available: one for the use with the Microsoft Xbox 360 game console and one for the use with a desktop computer. Both versions rely on the same software algorithm and have similar environmental restrictions:

- they must be used indoor,
- the user(s) must abide by specific range restrictions (see Figure 3.1).

On the other hand, the system can deal with:

- dark environments and changing light conditions,
- people moving through the background,
- occlusion of body parts by users and objects,
- leaving and re-entering the sensor area,
- different body sizes and shapes, skin colors and clothes.

To provide these features, hard and software must work together closely in a fine-tuned processing pipeline.

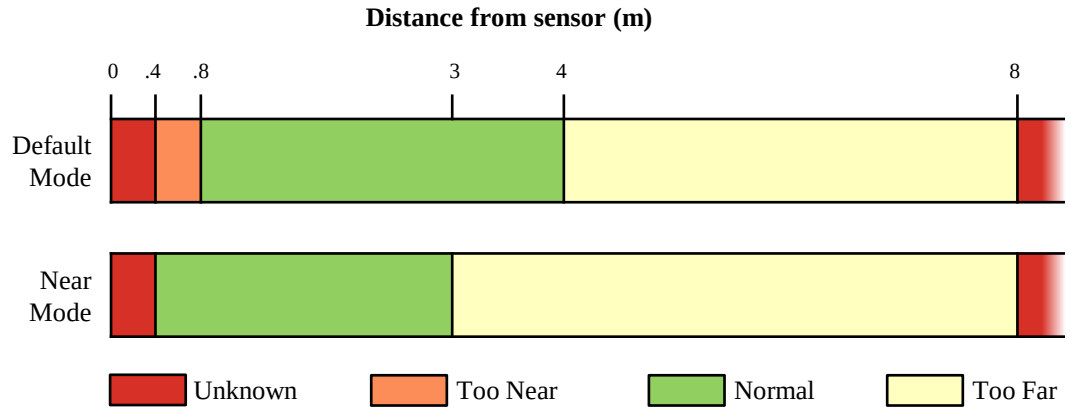


Figure 3.1: Range restrictions of the Kinect sensor (compare to [Kin12]). The "Near Mode" is only supported by the desktop version of the sensor.
For ranges marked with **Unknown**, no distance estimate is available. For ranges marked with **Too Near** and **Too Far**, estimates are made, but might be inaccurate.

A schematic overview over the pipeline is given in Figure 3.2. The steps which are vital for the high pose estimation performance, the steps one, three and four, are analyzed further in the following sections. Section 3.2 focuses on the hardware of the system, explaining how the depth image generation works and what information the hardware delivers to the pose estimation algorithms. With these explanations it becomes clear, why the aforementioned restrictions exist. In Section 3.3, the algorithms used to process the depth image and to do the pose estimation are explained in detail. Shotton et al. only provide information on steps three and four in their work [SFC+11].

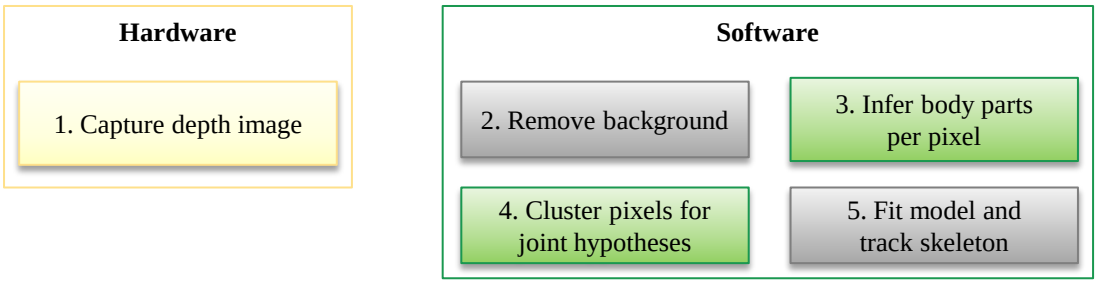


Figure 3.2: The processing pipeline of the Kinect pose estimation system (compare to [SFCB11]). The steps one, three and four are discussed in this chapter.

3.2 Depth Sensing

The Kinect sensor, providing the capabilities for the first processing step of capturing the depth image, has been developed by PrimeSense and is licensed by Microsoft. After the release of the device, iFixIt presented a teardown [iFi12] and analysis of the hardware. The sensor contains all major parts of a computer, from a mainboard with cooling system, 64Mb ram and flash memory to the imaging technology with an infrared projector and two CMOS cameras, amongst other parts. The core of the device is the PrimeSense image processor. The overall concept is illustrated in Figure 3.3 and Figure 3.4.

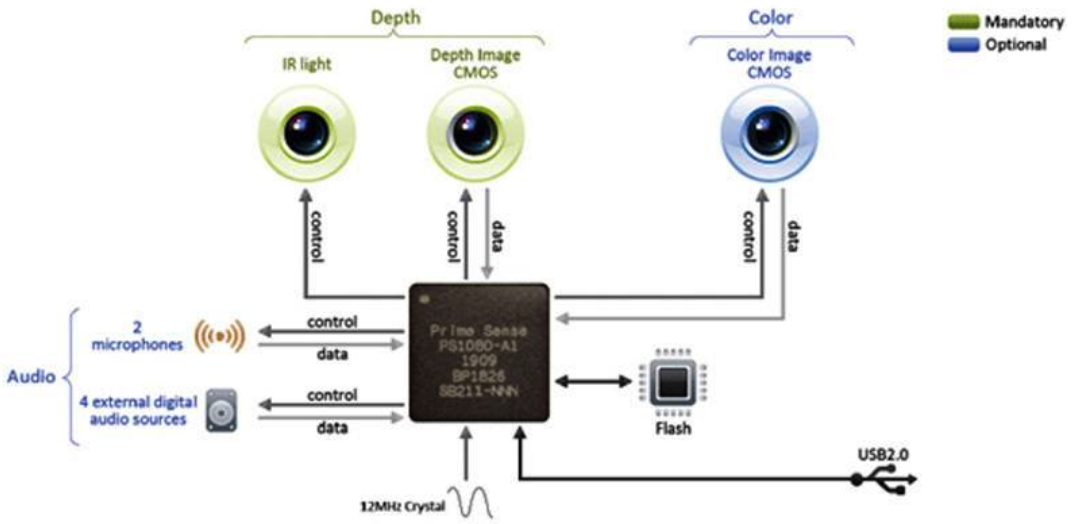


Figure 3.3: Hardware concept of the Kinect sensor [Pri12b]. Note, that the arrows to and from the "4 external digital audio sources" suggest, that this label is wrong. "audio sources" means speakers, but the depicted data and control flow implies audio sensors.

In Figure 3.3, the main hardware components and their functional connections are shown. The very left hand side of the graphic shows audio components that will not be discussed further in this work. Memory access is possible for the processor with a flash chip. The USB-connection allows for transmission of data to and from the connected host computer. Three components are responsible for the visual sensing: a pair of an infrared light projector and a CMOS sensor for depth images, and one CMOS sensor for standard color images.

How the depth image transmitter/receiver pair works together is illustrated in Figure 3.4. The infrared projector projects a pattern into the scene in front of the sensor and the CMOS sensor takes images of the scene with the projected pattern. The processor combines the knowledge of the projected pattern with the captured CMOS image to calculate the depth image. The process of depth estimation using this pattern is a patented technology by PrimeSense called LightCoding.

Figure 3.4: Depth sensing concept of the Kinect sensor [Pri12a].

LightCoding The infrared projector projects a pseudo-random speckle pattern into the room in front of the device. The pattern is characteristic for each sensor device and fixed during the assembly process. An example image of such a pattern can be found in Figure 3.5.

The figure shows a Kinect device (bottom of the image) and the projected pattern on a flat white surface. The arrangement of the blue points slightly shows a 3×3 checkerboard pattern with nine tiles, with a brighter dot in each tile center. A pincushion distortion effect is visible due to the projection properties of the lens.

This image is captured by the dedicated depth image CMOS sensor, as shown in Figure 3.4. The processor of the Kinect sensor calculates the depth of all the pattern speckles. This is done using the intrinsic and extrinsic camera and projector parameters and the knowledge of the pattern. With the support of the center points of each tile and the knowledge of the tile pattern, the depth of the other points within a tile can be inferred quickly by using depth information from the neighboring points [Via11, p. 11]. As to our knowledge, more information about the depth inference process is not available from public sources.

The aforementioned environmental restrictions originate from the speckle pattern projection: outdoor, the sunlight is strong enough that the infrared speckles can not be found by the sensor [Via11, p. 23]. The range restrictions are due to perspective distortion of the pattern. For larger distances, the speckles are further apart from each other, resulting in an unacceptable sampling density and bad depth estimate.

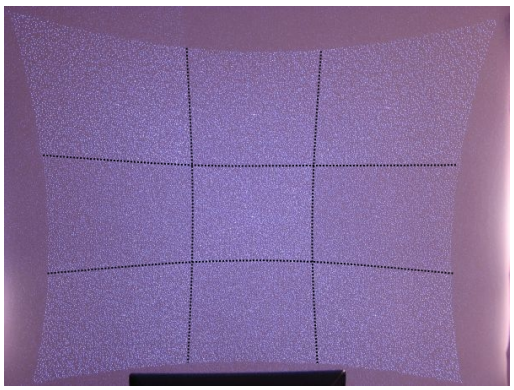


Figure 3.5: Speckle pattern projection of the Kinect hardware [fut11] with marked tile borders (black dotted lines). The borders were added to the image manually, because they are hardly visible on the printed image. The brighter dots in each tile center are not visible in the printed image.

All mentioned calculations are done by the processor on the Kinect sensor device and ready-to-use depth images with a resolution of 640×480 pixels are transferred at a rate of 30 Hz to the host computer. The accuracy of the depth estimation (approximately 0.4 cm at 1 m distance up to 5 cm at the maximum distance of 4 m [Via11, p. 16]) is sufficient to capture characteristic body part shapes.

3.3 Pose Estimation from Depth Images

The first step carried out by the software when receiving the depth images is subtracting the background and creating a user mask. The result of this preprocessing step is an image mask with one ID for each pixel (for an example, see Figure 3.6). This ID can either be a background flag or a user label, stating for each pixel whether it belongs to the background or to which user. This important preprocessing step is not described in [SFC+11] and no further information could be found on how it is implemented in the Kinect system.

The pose estimation algorithm gets the depth image and the background/user-label map per frame. Thus, the task of Human Pose Estimation can be broken down into the following main steps:

1. a per-pixel classification task to identify body parts (step 3 in Figure 3.2),
2. a clustering of pixels to eliminate noise and to estimate the joint positions (step 4) and finally
3. a model fitting and tracking step to track the body joint positions over time (step 5).

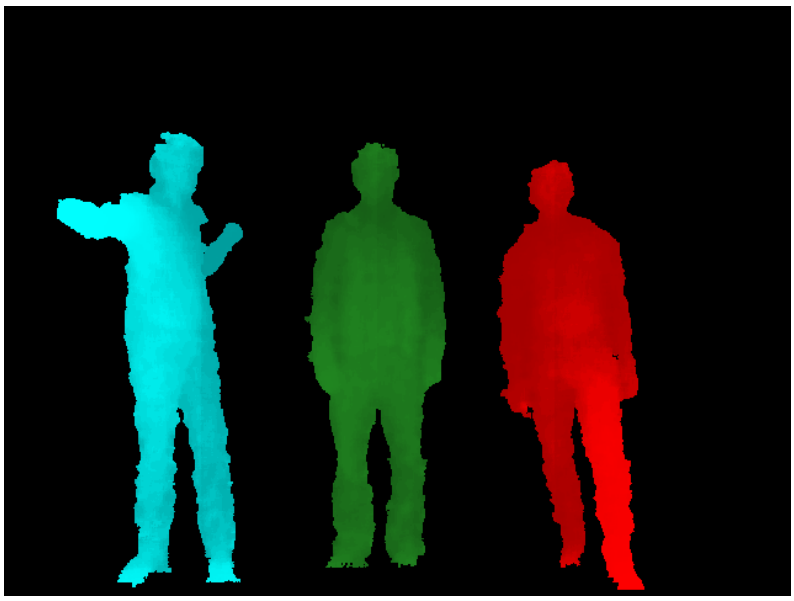


Figure 3.6: Input image to the pose estimation algorithms. For each pixel, the estimated depth and a label is available, labeling the pixel with a user ID or a background flag (here: **black -** background, **blue/green/red shade -** user labels). The depth information is illustrated with different brightness levels.

3.3.1 Pixel Classification

For the task of pixel classification, the human body is split into 31 parts of interest (see Figure 3.7). Some of them represent skeletal joints, the others fill the gaps or can be used in combination to predict body part positions, e.g. the four body part classes of the head (left upper, left lower, right upper, right lower). Let $\bar{C}$ be the set of body part classes. To classify a pixel as one of the body part classes $\bar{c} \in \bar{C}$, a Decision Forest trained on a set of discriminative depth comparison features is used.

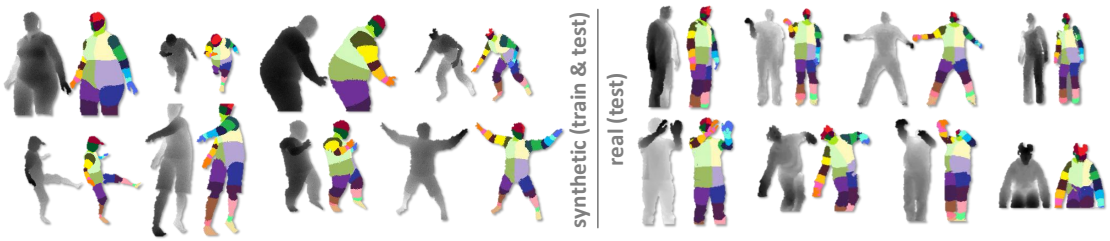


Figure 3.7: Pairs of depth images and images with body part classes [SFC+11, p. 3].

3.3.1.1 Depth Image Features

In the following paragraphs the variables k, l and m represent vectors, whereas x, y and z represent scalar coordinates. Scalar variables marked with a hat represent coordinates in the camera coordinate system (the coordinate system of the Kinect sensor), scalar variables without a hat represent coordinates within the image coordinate system.

The depth image features are easy to understand and can be computed very efficiently. For the following algorithm a set of features is used, originating from the parameterized meta-feature f. This feature is calculated at a feature extraction point k, consisting of the coordinates in image I, and has a parameter $\theta = (l, m)$. The parameter consists of the two offsets l and m, each specifying a two-dimensional offset from pixel k to two measurement points. The range of l and m is specified manually and defines the feature set size. With the parameter θ, the meta-feature represents a set of many features that is managed by the classifier.

The feature value is calculated as the difference of the depth values at the two measurement points as given in the following equation:

$$f_\theta(I, k) = d_I\left(k + \frac{l}{d_I(k)}\right) - d_I\left(k + \frac{m}{d_I(k)}\right). \tag{3.1}$$

In this equation, $d_I(k)$ is the depth value in image I at position k. The normalization step of dividing l and m by the depth of the feature extraction point is necessary to approximate depth invariance for the feature value calculation.

Depth Invariance Approximation To explain how the approximation works, it is necessary to have a look at the perspective transformation of the point coordinates in the depth image (for a short derivation of the transformation function, see Section 2.1.1). A point with camera coordinate x is located in the image at

$$x = f \cdot \left(\frac{\hat{x}}{\hat{z}}\right), \tag{3.2}$$

where f is the focal length of the camera and $\hat{z}$ the distance of the point to the camera. The feature measurement point for the depth image has an offset o from x in the image, so it is read at position $(x + o)$ in the image. In world coordinates, this refers to the depth at the point

$$(x + o) = f \cdot \frac{(\hat{x} + \hat{o})}{\hat{z}} \tag{3.3}$$

for a specific value $\hat{o}$.

The idea behind the feature calculation formula is, that $\hat{o}$ should remain constant in world-coordinates, so that it captures object specific depth-differences of the object shape. To keep $\hat{o}$ constant in Equation 3.3, o must be calculated dynamically, as derived in Equation 3.4:

$$\begin{aligned} o &= f \cdot \frac{(\hat{x} + \hat{o})}{\hat{z}} - x = \\ &= f \cdot \frac{(\hat{x} + \hat{o})}{\hat{z}} - f \cdot \frac{\hat{x}}{\hat{z}} \\ &= f \cdot \frac{\hat{o}}{\hat{z}} \end{aligned} \tag{3.4}$$

Since f and $\hat{o}$ are constant, they can be combined in one value that must be divided by $\hat{z}$, the depth of the feature extraction point. l and m in Equation 3.1 are such combined values. This method is just an approximation for depth invariance, because the real perspective transformation function has some more parameters than the used approximation in Equation 3.2 and Equation 3.3. The equations shown here for the x-coordinates are applicable for the y-coordinates accordingly.

(a) **(b)**

Figure 3.8: Example of the depth feature extraction for two features f_{θ_1} and f_{θ_2} at different extraction points for each feature [SFC+11, p. 4].

Figure 3.8 shows an example of feature extraction points (marked with a yellow x) and the according measurement points (marked with red circles) for two features f_{θ_1} and f_{θ_2} of the feature set. In the two subfigures, different extraction points are shown. If a measurement point lies on background or outside of the image, a large positive constant value is defined as result of the measurement.

f_{θ_1} is "searching" for big differences in the vertical direction and is of great help for e.g. classifying pixels in the upper body. This is visible in Figure 3.8, as f_{θ_1} has a large positive value in subfigure (a) and not in subfigure (b). f_{θ_2} in contrast "searches" for differences in the horizontal direction with a smaller offset and thus is very useful for the classification of e.g. arm pixels.

All the features of this feature set can be computed very efficiently. For the calculation, it is necessary to "read out at most 3 [depth] image pixels and perform at most 5 arithmetic operations; and the features can be straightforwardly implemented on the GPU" [SFC+11, p. 4].

3.3.1.2 Classification with Randomized Decision Forests

The initial set of depth features is very big at learning time, therefore for fast classification just the best suited features must be selected (i.e. to extract only few at evaluation time). Randomized Decision Trees and Randomized Decision Forests are used for this task.

The Decision Trees process examples as described in Section 2.3.2.1, with a feature f_θ and a threshold τ for each split node (the attribute "randomized" only refers to the way they are built). To classify the pixel with index k in image I with one tree, the distribution $P(\bar{c}|I,k)$ is stored in the leaf nodes of the tree.

The Decision Forest consists of many Decision Trees. To get the classification of the Decision Forest consisting of T Decision Trees, the classification result of each tree t is obtained, the distribution $P_t(\bar{c}|I,k)$. The distributions of all trees are averaged to get the final classification probabilities

$$P(\bar{c}|I,k) = \frac{1}{T}\sum_{t=1}^{T} P_t(\bar{c}|I,k). \tag{3.5}$$

The classification result is determined by choosing the most probable class

$$\arg\max_{\bar{c}\in\bar{C}} P(\bar{c}|I,k). \tag{3.6}$$

Training To train the Decision Trees efficiently with a high amount of training images, a special training strategy is used to avoid precalculating a large set of features for every image pixel. In order to reduce the amount of pixels used, "[...] a random subset of 2000 example pixels from each image is chosen to ensure a roughly even distribution across body parts" [SFC+11, p. 4]. This set of examples is $Q = \{(I_\iota, k_{\iota,\kappa})\}$, $\kappa \in [1;2000]$, I_ι standing for the ιth image and $k_{\iota,\kappa}$ representing the κth selected pixel in image ι.

An inductive inducer is used to build the decision trees from the given set of examples Q. A tree is considered to be "empty" at the beginning, and is built with the following algorithm (compare to [SFC+11, p. 4]):

1. Propose a set of s random splitting candidates $S = \{\phi_i = (\theta_i, \tau_i)\}$, $i \in [1;s]$. The splitting candidate ϕ_i consists of a feature θ_i and a threshold τ_i. Shotton et al. do not explain the random choice for the splitting candidates in detail, but it can be assumed that only the feature θ_i is selected randomly and the threshold τ_i is then calculated optimally from the training examples.
2. Partition the set of examples Q into left and right subsets by each ϕ_i:

$$\begin{aligned} Q_{left}(\phi_i) &= \{(I,k)|f_{\theta_i}(I,k) < \tau_i, (I,k) \in Q\} \\ Q_{right}(\phi_i) &= Q \backslash Q_{left}(\phi_i) \end{aligned} \tag{3.7}$$

3. Compute the ϕ giving the largest gain in information:

$$\phi^* = \arg\max_{\phi \in S} \text{Gain}(Q, \phi) \tag{3.8}$$

$$\text{Gain}(Q, \phi) = \text{Entropy}(Q) - \sum_{side \in \{left, right\}} \frac{|Q_{side}(\phi)|}{|Q|} \text{Entropy}(Q_{side}(\phi))$$

where the Shannon entropy $\text{Entropy}(Q)$ is computed on the normalized histogram of body part classes for all $(I, k) \in Q$.

4. If the largest gain $G(\phi^*)$ is sufficient, and the depth in the tree is below a maximum, add the split node ϕ^* to the decision tree. Then recurse for the left and right subsets $Q_{left}(\phi^*)$ and $Q_{right}(\phi^*)$.

The setup used for classification in the Kinect system consists of 3 trees with a depth of 20. This selection is determined with several experiments, as described in Section 3.4.

3.3.2 Joint Position Estimation

Having classified each pixel to one of the body part classes $\bar{c} \in \bar{C}$, it is now necessary to estimate the skeleton joint locations of the "joints" head, neck, left/right shoulder, left/right elbow, left/right hand, left/right hip, left/right knee, left/right foot. Some of these skeleton joints have a direct corresponding body part class, e.g. each elbow skeleton joint and the elbow classes, visible in Figure 3.7. The others can be treated specifically with the proposed method.

To find the skeleton joint positions from the pixel classes, a probability density function in 3D world space is generated for each skeleton joint class $c \in C$, with C being the set of skeleton joint classes. The Mean Shift Algorithm [CM02] is used to find modes in that distributions.

The density function for joint c is estimated as

$$f_c(\hat{k}) \propto \sum_{i=1}^{N} \overbrace{P(c|I, k_i) \cdot d_I(k_i)^2}^{w_{ic}} \cdot e^{-\left\| \frac{\hat{k} - \hat{k}_i}{b_c} \right\|^2}. \tag{3.9}$$

In this equation,

- $\hat{k}$ is a position in 3D world space,
- N is the number of image pixels,
- w_{ic} is a pixel weight, and is explained in the next paragraph,
- k_i is an image pixel,
- $\hat{k}_i$ is the reprojection of the image pixel k_i into 3D world coordinates,

- b_c is the bandwidth variable, learned for each joint class.

$P(c|I,k)$ used to calculate the pixel weight can be aggregated over body part classes relevant for the same skeleton joint position, e.g. for the four body part classes for parts of the head (left upper, left lower, right upper, right lower). The weight component incorporates the pixel depth to ensure that the density estimates are depth invariant. This gives "[...] a small but significant improvement in joint prediction accuracy" [SFC+11, p. 5].

The detected modes by the Mean Shift Algorithm lie on the body surface due to the definition of the density function in Equation 3.9: the Euclidean Norm in the exponent is always bigger for pixels outside the body surface, and with a negative exponent this causes lower probability values. Thus, pixels not on the body surface are never detected as modes by the Mean Shift Algorithm.

Because the modes lie on the body surface but the skeleton joints should be located at coordinates within the body, each detected mode is pushed back into the scene by a learned offset ς_c to produce a final joint position proposal. "The bandwidths b_c, probability threshold λ_c, and surface-to-interior z offset ς_c are optimized per-part on a hold-out validation set of 5000 images by grid search. (As an indication, this resulted in mean bandwidth 0.065m, probability threshold 0.14, and z offset 0.039m)" [SFC+11, p. 5].

The resulting joint position proposals can be passed to a final processing system, which realizes the last steps in Figure 3.2, fits the skeleton model and tracks the user over subsequent frames. The pose estimation system is designed to rely on per-frame information only, to be robust against lost joints and users leaving the scene. Even with no tracking algorithm employed, the results of the pose estimation algorithm are changing smoothly over multiple frames [SFCB11].

3.3.3 Data Generation

To avoid overfitting and to provide training examples with the aspired variability regarding shapes and poses, a mixture of real and synthetic training data is used. The real training data is acquired by capturing frames with the depth camera and hand-annotating them afterwards.

To generate artificial data reflecting realistic poses, a database with motion capture data is created, containing about 500,000 poses in sequences of many activities. The motion capture data does not contain depth images, just 3D positions of every joint. In a second step, the motion capture data is retargeted to fifteen 3D human models, varying in body size and shape. Further random variations are generated by slightly varying height and weight of the models and also camera pose, camera noise, clothing and hairstyle. All body parts of the models are textured by hand once. The ground truth (depth map with body part annotations) is obtained by rendering the motion capture poses for any model (see Figure 3.7).

To reduce the motion capture data to the most important poses for training, similar poses (e.g. captured during a slow movement) are sampled. In order to do this, a 'furthest neighbor' clustering is done, with the maximum euclidean distance between corresponding body joints as distance measure between poses. For each cluster, just one representative pose is used. This breaks down the pose training set to a size of 100,000 poses with a minimum distance of 5 cm. The pose database is refined repeatedly to fill the "gaps" in the pose space that seem to be underrepresented after classifier training.

3.4 Results

3.4.1 Evaluation

In [SFC+11], the system is evaluated in many experiments with different datasets. It is compared with the former best-performing method by Ganapathi et al. [GPKT10] on their test dataset taken with a time-of-flight camera. A significant precision improvement is shown, with a runtime improvement of a factor greater than ten (see Figure 3.9). This could be achieved even though the test data is taken with a camera with different properties and no use of tracking information, in contrast to [GPKT10].

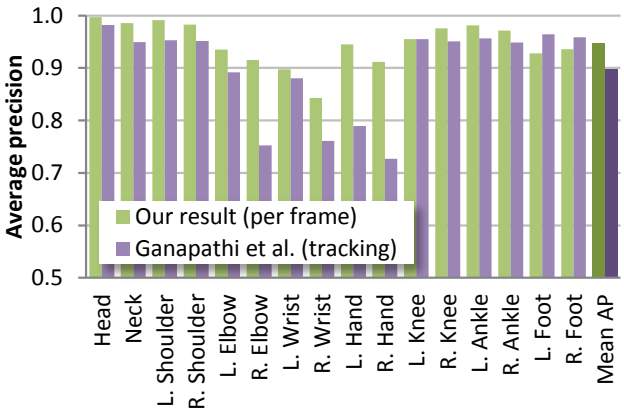


Figure 3.9: Comparison of the approaches [SFC+11] and [GPKT10], [SFC+11, p. 8]. "Our result (per frame)" refers to the approach by Shotton et al. [SFC+11].

An important part of the discussion in [SFC+11] deals with the paid loss of precision due to the pixel classification step before the joint detection. Figure 3.10 shows the resulting precision of the joint proposal on ground truth 3D images and on automatically analyzed images with the Decision Forest. The mean average precision difference is approximately 18%. However, the authors highlight the need of using body parts for the HPE task, by showing that a whole-body classification approach

using nearest-neighbor algorithms needs a much higher amount of training images to reach comparable accuracy, while needing more processing time and working on a restricted, easier classification task during this test [SFC+11, p. 7].

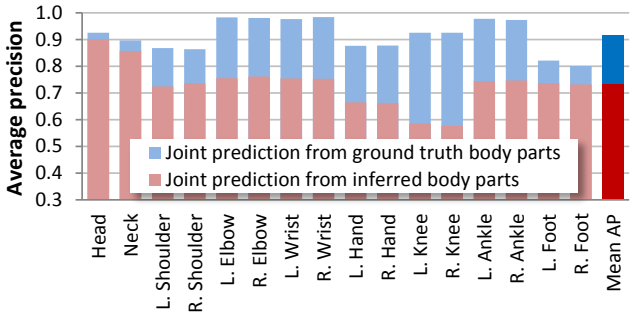
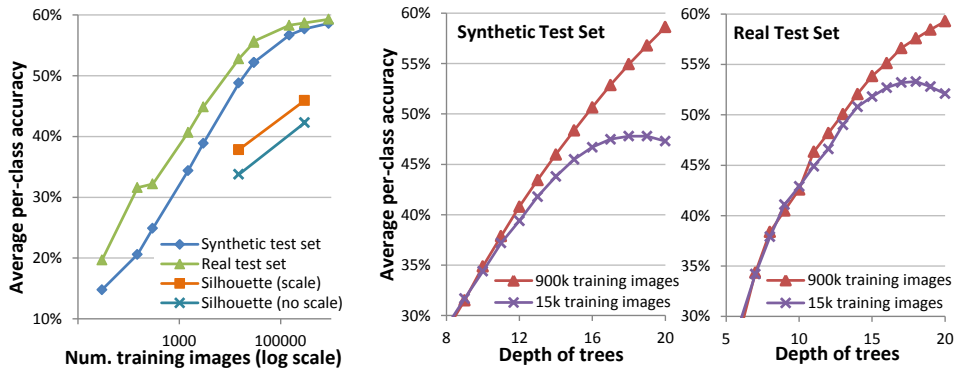


Figure 3.10: Joint prediction accuracy from pixel labels and ground truth labels [SFC+11, p. 7].

Figure 3.11 shows three charts about the impact of variation in training parameters on classification accuracy. The trends in all three graphics for synthetic and real test sets are highly correlated. This speaks for a successful synthetization of data. The real test set even seems to be 'easier' than the synthetic one. This can be explained with the additional added variability in the synthetic data.



(a) Number of training images. **(b)** Depth of trees.

Figure 3.11: Training parameters vs. classification accuracy [SFC+11, p. 6].

Figure 3.11a shows the average per-class accuracy per number of training images on a log scale. The accuracy continually improves, but the improvement slows down at approximately 100,000 training images. The authors of [SFC+11] suspect this to be caused by the limited model capacity of the used 3 tree Decision Forest with a tree depth of 20.

In the same figure, there is also the precision given for the classification of silhouette images. For these images, just the difference of the flag between person and background is available. The curve with "(scale)" was generated allowing the features to

scale with the depth of the feature extraction point as explained in Section 3.3.1.1, with the average depth of the silhouette as extraction point depth. The experiment for the curve with "(no scale)" was conducted without allowing this. "For the corresponding joint prediction using a 2D metric with a 10 pixel true positive threshold, we got 0.539 mAP [mean average precision] with scale and 0.465 mAP without. While clearly a harder task due to depth ambiguity, these results suggest the applicability of our approach to other imaging modalities" [SFC+11, p. 6].

Figure 3.11b shows the accuracy of the joint localizations per depth of the trees of the pixel classifier and two training set sizes. Until a depth of ten, the improvement on both test sets is high, independent of the training set size. This suggests an efficient training strategy.

At higher depths of trees, the curves for the 15,000 image sized training set starts to tail off, with overfitting being observed for depths 17 or higher. Overfitting does not occur for the training set with 900,000 images until a tree depth of 20, but the improvement rate starts to decrease for the depths at about 20. "Of all the training parameters, the depth appears to have the most significant effect as it directly impacts the model capacity of the classifier" [SFC+11, p. 6].

3.4.2 Discussion

The presented approach proves that Human Pose Estimation can be solved reliably in super real-time in a highly variable scenario by using depth information.

The use of a large parameterized feature set based on a simple meta-feature enables the assembly of a capable classifier by using a Decision Forest to manage the features. The power of the simple features lies in their multitude and variety, and the ability of the Decision Trees to choose the important features for classification.

The results show clearly the increasing need for example images with increasing model learning capability of the classifier. The great complexity of the Human Pose Estimation task requires a high learning capability and thus many examples. The synthetization of additional training and test data with depth images can provide the necessary amount of images. Unfortunately, this great possibility cannot be straightforward extended to color images, since no model exists to capture the high variability in this scenario. To do so, an image representation not based on single pixels must be found, so that it does not get "confused" by local color anomalies.

4 Multiple Pose Context Trees for estimating Human Pose in Object Context

An object interaction is defined by Singh et al. as an activity for which a visible object is approached with a body part, usually a limb. Examples are playing soccer or basketball, with the soccer ball or the basket ball as interaction objects. Singh et al. show in their work "Multiple Pose Context Trees for estimating Human Pose in Object Context" [SKN10] how such interactions can be modeled and detected in 2D color images. The detection involves pose estimation of the interacting person, object detection and recognition of the interaction. Singh et al. show that pose estimation can be improved by taking into account the additional contextual information of the interaction models.

4.1 Motivation and Concept

Human Pose Estimation can largely benefit from incorporating interactions with a visible object in the pose estimation process. When an interaction object is detected in an image, it gives a clue that e.g. an interacting limb could be nearby. When a HPE system must decide between multiple hypotheses for the position of that limb it can make use of this clue to make the right selection. However, to do this, the system must have detectors for all possible interaction objects and models of all interactions, telling it which body part interacts with which object.

Singh et al. propose for this purpose a new graphical model: Pose Context Trees (PCTs). For each known interaction type, the information about interaction and pose is encoded in one Pose Context Tree.

A Bayesian framework is used to infer "[...] the optimal pose-object pair by maximizing the likelihood over multiple Pose Context Trees for all interactions" [SKN10, p. 1]. The proposed method can deal with unknown interactions by falling back to a standard pose estimation method. For an example on how knowledge about object interaction can improve pose estimation, see Figure 4.1.

In the following section, the concept of Pose Context Trees is explained. They are used to create a model for each object interaction. Section 4.3 explains how the

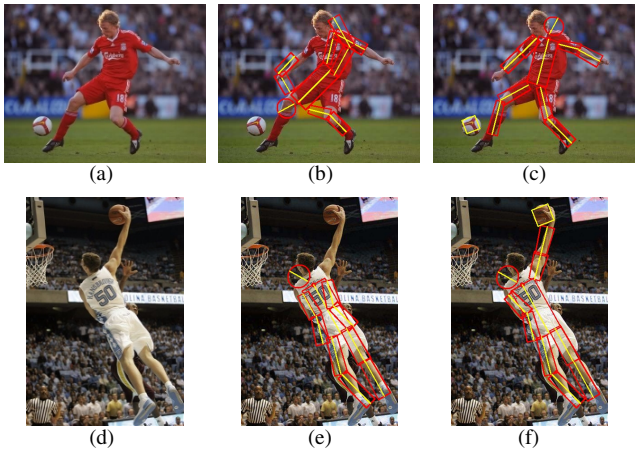


Figure 4.1: Example series of images showing the benefit for pose estimation from knowledge about interaction context [SKN10, p. 1]. **Top row:** images of a soccer player, **bottom row:** images of a basketball player. (a) and (d) are the original images, (b) and (e) show the result of pose estimation without including information on the interaction object. (c) and (f) show the result of pose estimation using the proposed approach, including the interaction object in the model matching process. A significant improvement of the estimated pose is achieved.

trees are used for matching and how the interaction can be inferred. The chapter closes with an evaluation and discussion of the approach in Section 4.4.

4.2 Pose Context Trees

The Pose Context Trees introduced by Singh et al. capture information about possible poses and the interaction with the interaction object. The basic modeling steps are the same as for Pictorial Structure Models. An introduction to Pictorial Structure Models can be found in Section 2.3.4.

4.2.1 The Body Model

Singh et al. model the human body with the PSM introduced in Section 2.3.4.5. The only difference is that they do not include the scale of detected body parts into the inference process. Due to the high similarity to the model described in Section 2.3.4.5, variables and terms are only briefly introduced in this section. For more details and a derivation of the matching framework, see Section 2.3.4.

The human body is modeled as a combination of ten body parts of interest (see Figure 2.10). It has a tree structure with the torso as root node and the head and the four limbs as branches.

Formally, the model (V, E) consists of vertices V and edges E. Each node in the model is denoted as shown in Figure 2.10 and connected to an observation node. The observation nodes represent the observed values for the configuration of each body part, with an entire body configuration denoted by $L = \{l_i\}_{v_i \in V}$. The difference to the model in Section 2.3.4.5 is that a body part configuration is given by $l_i = (x_i, y_i, \theta_i)$, consisting only of the location and rotation of the body part.

The model parameters $\Theta = (u, E, c)$ consist of the appearance parameters $u = \{u_1, \ldots, u_n\}$, the set of edges E and the connection parameters c. In this approach, the set of edges E is regarded as given (as shown in Figure 2.10) and is not learned from training data, thus it can be disregarded in the following explanations. The edges contain the connection parameters $c = \{c_{ij} | (v_i, v_j) \in E\}$.

With these declarations, the statistical framework for the body model is formulated as explained in Section 2.3.4:

$$\begin{aligned} P(L|I,\Theta) &\propto P(I|L,\Theta) \cdot P(L|\Theta) \\ &= \underbrace{\prod_{v_i \in V} P\left(I|l_i, u_i\right)}_{\text{Appearance terms}} \cdot \underbrace{\prod_{(v_i,v_j) \in E} P\left(l_i, l_j | c_{ij}\right)}_{\text{Prior terms}}, \end{aligned} \tag{4.1}$$

with the body part appearance terms $P\left(I|l_i, u_i\right)$ and the prior terms $P\left(l_i, l_j|c_{ij}\right)$ for each connection. The values of the appearance terms are determined by using the body part detectors described in Section 4.3.3; the appearance parameters contain the configurations for these detectors.

The prior terms are not fully defined by Singh et al. To be able to use the efficient matching algorithms described in [FH05, Ram07], Singh et al. assume that they are estimated by a connection function in a specific form. This form is a Diagonal Gaussian Normal Distribution over the differences of the body part configurations, i.e.

$$P\left(l_i, l_j|c_{ij}\right) = \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}), \tag{4.2}$$

where T_{ij}, T_{ji} are transformation functions and D_{ij} is a diagonal covariance matrix. The meaning of the transformation functions T_{ij}, T_{ji}, of the connection parameters c_{ij} and the matrix D_{ij} is explained in Section 2.3.4. Singh et al. do not specify their choice for the transformation functions T_{ij} and T_{ji}. It can be assumed that they use a standard definition for HPE, similar to the one described in Equation 2.25 but omitting the scale dimension.

This framework gives a matching probability that can be maximized by approaches given in [FH05, Ram07]. It is extended in the following section to allow for the modeling of object interaction.

4.2.2 Extension to Pose Context Trees

In addition to the body parts, an interaction a and the interaction object $O(a)$ is modeled with an additional node in the graphical model (e.g. a could be "soccer" and $O(a)$ the soccer ball). To enable for efficient inference, it is favorable to preserve the tree structure of the model. This means that the new object node must be connected exactly to one node v_k in the model tree. This is possible for most interactions, since "[...] humans often interact with scene objects with part extremities (legs, hands) [...]" [SKN10, p. 2]. Thus, an interaction can often be modeled by connecting the interaction object node to one limb leaf node (similar to Figure 4.2).

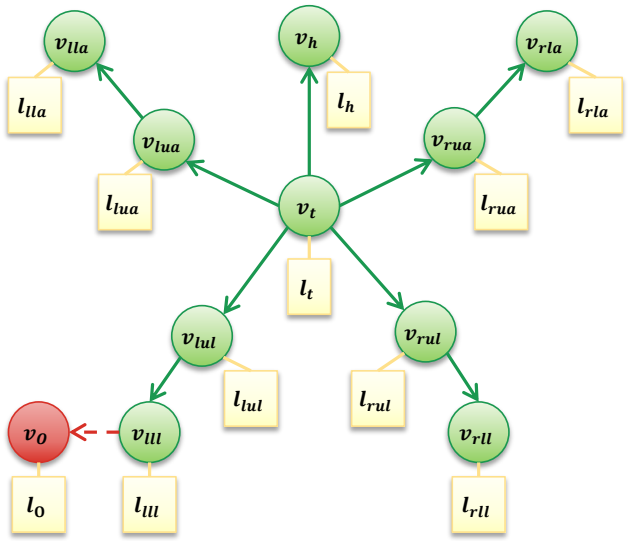


Figure 4.2: Pose Context Tree for an object interaction with the left lower leg. The added object node v_o is marked red.

In this section, the Pose Context Tree model for interaction a is described. During these explanations, let for simplicity O denote the interaction object for interaction a, i.e. $O \equiv O(a)$.

The new object node v_O is part of the extended node set of the PCT V_O, i.e. $V_O = V \cup \{v_O\}$. Let the extended configuration be $L_O = L \cup \{l_O\}$ and the extended appearance parameters $u_O = u \cup \{u_O\}$, now containing the configuration l_O and the appearance parameters u_O of the interaction object. To fully define the extended model parameters Θ_O, the connection parameter and edge set are defined as $c_O = c \cup c_{kO}$ and $E_O = E \cup \{v_k, v_O\}$ respectively. With these definitions $\Theta_O = (u_O, E_O, c_O)$.

The extended model is still a PSM and its matching probability is captured by the

Bayesian framework:

$$\begin{aligned} P(L_O|I,\Theta_O) &\propto P\left(I|L_O,\Theta_O\right)\cdot P\left(L_O|\Theta_O\right) \\ &= \prod_{v_i\in V_O} P\left(I|l_i,u_i\right)\cdot \prod_{(v_i,v_j)\in E_O} P\left(l_i,l_j|c_{ij}\right), \end{aligned} \tag{4.3}$$

with the standard modeling steps for PSMs. Singh et al. lead this formulation back to the matching framework of their human body model in Equation 4.1. This is a necessary step to show that the results of multiple PCTs based on the same human model are comparable. With the following transformations it is shown that they are proportional to the matching probability of the human PSM multiplied with two terms relevant for the interaction object:

$$\begin{aligned} P(L_O|I,\Theta_O) &\propto \prod_{v_i\in V_O} P\left(I|l_i,u_i\right)\cdot \prod_{(v_i,v_j)\in E_O} P\left(l_i,l_j|c_{ij}\right) \\ &= \overbrace{\left(\prod_{v_i\in V} P\left(I|l_i,u_i\right)\right)}^{P(I|L,\Theta)}\cdot P\left(I|l_O,u_O\right)\cdot \\ &\quad \underbrace{\prod_{(v_i,v_j)\in E} P\left(l_i,l_j|c_{ij}\right)\cdot P\left(l_k,l_O|c_{kO}\right)}_{P(L|\Theta)} \\ &= \underbrace{P\left(I|L,\Theta\right)\cdot P\left(L|\Theta\right)}_{\text{Body model: } P(L|I,\Theta)}\cdot P\left(I|l_O,u_O\right)\cdot P\left(l_k,l_O|c_{kO}\right) \\ &\propto P\left(L|I,\Theta\right)\cdot P\left(I|l_O,u_O\right)\cdot P\left(l_k,l_O|c_{kO}\right), \end{aligned} \tag{4.4}$$

where $P(L|I,\Theta)$ is the standard PSM probability for the body model defined in Equation 4.1. Only the two new terms $P(I|l_O,u_O)$ and $P(l_k,l_O|c_{ko})$ must be defined to fully specify the PCT.

The first additional term, $P(I|l_O,u_O)$, is an object appearance term and its value is estimated with an object detector (the object detectors used by Singh et al. are described in Section 4.3.3.2). The second term, $P\left(l_k,l_O|c_{ko}\right)$, is the prior term for the object and the interacting body part. Singh et al. model this probability as binary "box" prior

$$P(l_k,l_O|c_{kO}) = \begin{cases} 1 & |T_{kO}(l_k) - T_{Ok}(l_O)| < d_{kO} \\ 0 & \text{otherwise,} \end{cases} \tag{4.5}$$

where $|.|$ is the absolute norm for each vector component. With this definition, the prior defines an area around the connection points of the two parts in which the probability equals one. Again, Singh et al. do not specify the transformation functions. The choice of the connection function, its parameters and the fact that they

only model balls as objects (which are rotational invariant) for their experiments (see Section 4.4.1.2) lead to the conclusion that they ignore scale and rotation for defining it.

In this case, T_{kO}, T_{Ok} and d_{kO} can simply be defined by

$$\begin{aligned} T_{kO}(l_k) &= (x'_k, y'_k), \\ T_{Ok}(l_O) &= (x'_O, y'_O), \\ d_{kO} &= (d_x, d_y), \end{aligned} \tag{4.6}$$

where x'_k, y'_k, x'_O and y'_O are the image coordinates of the connection points of body part and interaction object. They can be determined by translating the connection points by the object detection points, i.e.

$$\begin{aligned} \begin{pmatrix} x'_k \\ y'_k \end{pmatrix} &= \begin{pmatrix} x_k \\ y_k \end{pmatrix} + \begin{pmatrix} x_{kO} \\ y_{kO} \end{pmatrix}, \\ \begin{pmatrix} x'_O \\ y'_O \end{pmatrix} &= \begin{pmatrix} x_O \\ y_O \end{pmatrix} + \begin{pmatrix} x_{Ok} \\ y_{Ok} \end{pmatrix}. \end{aligned} \tag{4.7}$$

This leads to the definition of the connection parameters c_{kO} as

$$c_{kO} = (x_{kO}, y_{kO}, x_{Ok}, y_{Ok}, d_x, d_y). \tag{4.8}$$

With these steps, the Pose Context Tree model is defined that captures all body parts, the interacting body part and the interaction object as well as the connection between them. With the definitions in the above paragraphs, the model can be matched efficiently with algorithms for matching PSMs. However, the question remains how the best fitting of many modeled interactions (i.e. multiple PCTs) can be determined. The solution of Singh et al. is explained in the next section.

4.3 Using Multiple Pose Context Trees

Each interaction that must be detected is modeled as a different PCT. However, all PCTs must be based on the same human PSM. Otherwise, the matching results of the PCTs would not be comparable, since the values of the probabilities for a model configuration are only estimated (note the "proportional to" sign in Equation 4.1). This way, several interactions for the same human model can be specified.

The final result of the algorithm is the best matching pose L^* and the most probable interaction a^*. To find them, it is necessary to include the Pose Context Trees for all modeled interactions in the inference process. Singh et al. approximate the mathematically exact solution and manage to build in a fallback option: when the algorithm encounters a non-modeled interaction it tries to estimate the pose without context information.

4.3.1 Mathematical Formulation

All modeled interactions a_i are in the set $\mathcal{A} = \{a_i\}_i \cup \{\phi\}$, where ϕ is a placeholder for unmodeled interactions. The model parameters $\Theta_{O(\phi)}$ are defined to be the model parameters of the basic human PSM, i.e. $\Theta_{O(\phi)} = \Theta$. Let $\mathcal{L}$ be the set of all possible configurations of the basic human PSM; then the optimal pose and interaction pair $(L^*, a^*) \in \{(L, a) | a \in \mathcal{A}, L \in \mathcal{L}\} = \Omega$ is defined as:

$$\begin{aligned}(L^*, a^*) &= \arg\max_{\Omega} P\left(L, a | I, \Theta_{O(a)}\right) \\ &= \arg\max_{\Omega} P\left(a | L, I, \Theta_{O(a)}\right) \cdot P\left(L | I, \Theta_{O(a)}\right) \end{aligned} \tag{4.9}$$

The conditional likelihood $P\left(a | L, I, \Theta_{O(a)}\right)$ of interaction a given the pose configuration L, image I and model parameters $\Theta_{O(a)}$ is modeled as the product of the likelihood of the corresponding object $O(a)$ in the neighborhood of L and the likelihood for the absence of objects that belong to other interactions in $\mathcal{A}$, i.e.

$$\begin{aligned}P\left(a | L, I, \Theta_{O(a)}\right) = \overbrace{P\left(l_{O(a)} | L, I, \Theta_{O(a)}\right)}^{\text{Prob. for the object}} \cdot \\ \underbrace{\prod_{a' \in \mathcal{A} \setminus \{a\}} \left(1 - P\left(l_{O(a')} | L, I, \Theta_{O(a')}\right)\right)}_{\text{Penalty terms}}.\end{aligned} \tag{4.10}$$

While $P\left(l_{O(a)} | L, I, \Theta_{O(a)}\right)$ is eliminated in the further derivation process, the probability $P\left(l_{O(a')} | L, I, \Theta_{O(a')}\right)$ must be calculated to obtain the penalty terms. Singh et al. do not specify how these probabilities are estimated. They can be modeled as the maximum probability estimate of an object detector in a specified neighborhood of the estimated pose L.

For shorter formulation, the penalty terms for pose L and interaction a are combined in $\Delta(L, a)$. Substituting $P\left(a | L, I, \Theta_{O(a)}\right)$ in Equation 4.9 with the result of Equation 4.10 leads to the final description of the optimal pose-interaction pair:

$$\begin{aligned}(L^*, a^*) &= \arg\max_{\Omega} P\left(l_{O(a)} | L, I, \Theta_{O(a)}\right) \cdot \Delta(L, a) \cdot P\left(L | I, \Theta_{O(a)}\right) \\ &= \arg\max_{\Omega} \underbrace{P\left(l_{O(a)} | L, I, \Theta_{O(a)}\right) \cdot P\left(L | I, \Theta_{O(a)}\right)}_{P\left(L_{O(a)} | I, \Theta_{O(a)}\right)} \cdot \Delta(L, a).\end{aligned} \tag{4.11}$$

The first two terms of the product can be combined to $P\left(L_{O(a)} | I, \Theta_{O(a)}\right)$, the matching probability of the PCT for interaction a. This probability on its own can be maximized efficiently using the standard solution strategies for PSMs. Singh et

al. do not take into the account the dependency on the penalty term $\Delta(L, a)$ and approximate the solution in a two step process.

In a first step, the best configuration for each PCT is determined, i.e. for each interaction $a \in \mathcal{A}$, the best pose $L^*_{O(a)}$ is estimated by matching the PCT. In the second step, the penalties are included for other detected objects close to $L^*_{O(a)}$.

"Observe that when [the best matching interaction] $a^* = \phi$, the problem reduces to finding the best pose given the observation and adds a penalty if objects are found near the inferred pose. Thus our model can be applied on any image even when the interaction in the image is not modeled, thereby making our model more apt for estimating human poses in general scenarios" [SKN10, p. 3].

4.3.2 Algorithmic Realization

The algorithmic approximation to the problem formulated above finds the best match for each PCT and penalizes the results with the penalty terms in a separate step. With $\mathcal{L}_{O(a)}$ being the set of all possible configurations of the PCT for interaction a, the algorithm is defined as follows (compare to [SKN10, p. 3]):

1. Match the Pose Context Trees for all interactions $a_i \in \mathcal{A}$ and obtain

$$\begin{aligned} P_i &= \max_{\mathcal{L}_{O(a_i)}} P\left(L_{O(a_i)} | I, \Theta_{O(a_i)}\right), \\ L^*_{O(a_i)} &= \arg\max_{\mathcal{L}_{O(a_i)}} P\left(L_{O(a_i)} | I, \Theta_{O(a_i)}\right). \end{aligned} \tag{4.12}$$

2. Extract the poses without the interaction objects from each Pose Context Tree. With $|.|_L$ denoting the projection on the body nodes, this can be formulated as:

$$L^*_i = \left| L^*_{O(a_i)} \right|_L . \tag{4.13}$$

3. Calculate the penalties:

$$\Delta_i = \prod_{a' \in \mathcal{A} \setminus \{a_i\}} \left(1 - P\left(l_{O(a')} | L^*_i, I, \Theta_{O(a')}\right)\right). \tag{4.14}$$

4. Select the pose-interaction pair with the highest score after taking the penalties into account:

$$(L^*, a^*) = (L_i, a_i), \; i = \arg\max_i P_i \cdot \Delta_i. \tag{4.15}$$

To match the Pose Context Trees and obtain the one with the highest matching probability in step one it is necessary to maximize the probability in Equation 4.4. Singh et al. use a special matching algorithm that updates the detectors during the matching process. The proposed algorithm is explained in Section 4.3.3.1.

4.3.3 Implementation

The matching algorithm works iteratively to improve the results from the detectors by switching between a color model and a shape model. The detectors for body parts and objects are created with very different methods, since they face different challenges. Whereas the object detectors in the analyzed approach mainly search for rigid objects with a very distinctive shape and color, the body parts can have high variation in both aspects.

(a) Edge template. **(b)** Region template.

Figure 4.3: Body part detection templates [SKN10, p. 4]. Light areas correspond to positive weights, dark areas to negative weights.

4.3.3.1 Body Part Detectors and PCT Matching

The body part detectors used by Singh et al. are based on the ones used by Ramanan in [Ram07]. Two templates are matched for each body part to calculate the detection response: an edge template and a part region template (see Figure 4.3). Singh et al. use the original templates provided by Ramanan [Ram07]. The body part detectors and the PCT are matched in a combined process to generate mutual benefits.

The algorithm for matching a PCT is sketched by Singh et al. as follows (see [SKN10, p. 4], [Ram07]):

1. In a pre-processing step, obtain the object likelihood map with an object detector as explained in Section 4.3.3.2.
2. Initialize the iterative process by correlating the edge template for each body part in Figure 4.3a with the Sobel edge map of the image. Store the results as body part likelihood maps.
3. Use the part likelihood maps and object likelihood map to estimate the pose with the Pose Context Tree and get a body configuration by using a message passing algorithm [Ram07].
4. From the currently known body configuration, learn an appearance model for each body part.

a) Create two RGB histograms $h_{\text{foreground}}$ and $h_{\text{background}}$ of the body part and of its background.

b) Calculate the appearance likelihood map of the body part as the binary map $P(h_{\text{foreground}}|I) > P(h_{\text{background}}|I)$.

c) Convolve this map with the body parts region template in Figure 4.3b to obtain a new body part likelihood map.

5. Obtain a new estimation of the pose with the new body part likelihood maps and compare it to the former estimated pose. If the termination criterion is met, return, otherwise go to step 3.

The termination criterion is not specified by Singh et al. They state, that in their "[...] experience, both the parses [estimated body part configurations] and the color models empirically converge after 1-2 iterations" [SKN10, p. 5].

Training As mentioned before, the templates for the body part detectors are used unchanged as provided by Ramanan in [Ram07] (for details on how they are learned, see [RS06]). However, the parameters for the connection functions of the body parts $c_{ij} \in c$ are learned from training data. "Given the annotations [...] available from the training data, we learn the Gaussian parameters with a Maximum Likelihood Estimator [FH05, ARS09]" [SKN10, p. 5].

4.3.3.2 Object Detectors

For each object class, a separate binary detector is used. The detector is trained using Gentle AdaBoost [FHT00] on Decision Trees with depth one or two (for an introduction to both techniques, see Section 2.3.3.3 and Section 2.3.2.1). A sliding window approach is used to obtain the object likelihood map.

The detectors rely on two feature types:

Object Outline The object outline is modeled with a variant of the HOG descriptor (for an explanation of this point descriptor, see Section 2.3.1). In the approach by Singh et al., the region of interest is split into rectangular cells and the histogram of oriented gradients is calculated for each cell. The histogram is then sum-normalized to 1. However, additional normalization steps and the soft binning seem to be omitted by Singh et al. (again, no explicit explanation is given). This is acceptable, since the objects that must be detected in the experiments by Singh et al. are round and invariant to rotation.

Object Color Additionally, for each cell, the sum-normalized histograms over hue and saturation in the HSV colorspace are constructed. The mean RGB and HSV values are used as additional descriptors per cell.

The final detection result is determined by extracting the features from the window and running them through the boosted Decision Trees.

Training For the training of each detector, annotated sample images from the internet are used. During training of the detector for one class, images from other object classes are added to the negative example set. To increase the robustness of the detector towards rotation and scale, the positive example set is expanded with rotated and scaled versions of positive examples.

"For each object detector, the detection parameters include number of horizontal and vertical partitions of the window, number of histogram bins for gradients, hue and saturation, number of boosted trees and their depths for the classifier, and were selected based on the performance of the detector on the validation set. [...] We select the classifier that gives [the] lowest False Positive Rate" [SKN10, p. 5].

This selection strategy is very important: false positive responses by the detectors can lead to misclassification of an image without an interaction object. Any object detection near a human will "pull" the match of the according PCT in its direction. That PCT is likely to receive the highest score and the estimated pose is strongly biased in direction of the non-existent object (see also Section 4.4.1.2).

The connection function for the objects is modeled as binary box prior (the function is defined in Equation 4.5, the parameters in Equation 4.8). The only parameters for this function are the connection points and the maximum difference allowed in each direction. The connection points are calculated as the mean difference between the object and interaction body part position in each dimension; the maximum allowed difference is defined as $\frac{1}{2}\sqrt{\sigma}$ of this difference [SKN10, p. 5].

4.4 Results

4.4.1 Evaluation

To evaluate this approach, a new dataset is assembled to have enough images with object interactions. Three interaction models are trained for the interaction categories *soccer*, *basketball* and miscellaneous (referred to as *misc*). The dataset contains "[...] images downloaded from the Internet and other datasets [GKD09, DT05, Ram07]. The dataset has 21-22 images for [the] 3 interactions - *legs with soccer ball*, *hand with basketball* and *miscellaneous*. [...] The *soccer* set includes images of players kicking or dribbling the ball, [the] *basketball* set has images of players shooting or holding the ball, and the *misc* set includes images from [the] People dataset [Ram07] and INRIA pedestrian dataset [DT05]" [SKN10, p. 5]. The algorithm is not evaluated on the dataset for human-object interaction recognition proposed by Gupta et al. [GKD09], because Singh et al. assume that the entire person is visible in the image. Many images in [GKD09] violate this assumption.

4.4.1.1 Object Detection

The object detection quality is evaluated per object type. A hypothesis is considered to be correct if the detection bounding box overlaps with the ground truth bounding box by more than 50%. For a list of the parameter configurations for the two object detectors, see [SKN10, p. 5]. The true positive rate and false alarm rate for each detector is shown in Table 4.1. A low false alarm rate is crucial for the success of this method, as discussed in the next section.

Detector	True Positive	False Positive
Soccer ball	91.7 %	$2 \cdot 10^{-4}$ %
Basket ball	84.2 %	$1.5 \cdot 10^{-3}$ %

Table 4.1: Object detection accuracy for the dedicated object detection test set [SKN10, p. 6].

4.4.1.2 Pose Estimation

The pose estimation accuracy is compared to [Ram07] (Method A), which uses nearly the same pose estimation approach without including context information. For the analyzed approach by Singh et al., accuracy is calculated assuming the interaction type is known (Method B, KnownObject-PCT), and once using the fully automatic approach inferring the interaction type (Method C, MultiplePCT). Thus, method B gives an upper bound for Method C, as it shows the pose estimation results for correctly inferred interactions.

"Pose accuracy is computed as the average correctness of each body part over all the images (total of 650 parts). An estimated body part is considered correct if its segment endpoints lie within 50% of the length of the ground-truth segment from their annotated location, as in earlier reported results" [SKN10, p. 6]. Table 4.2 shows the achieved estimation accuracy by the three methods.

Method	Pose Estimation accuracy (in %)			
	Soccer	Basketball	Misc	Overall
A (Pose estimation only, [Ram07])	67.1	43.2	64.3	57.33
B (KnownObject-PCT)	72.9	63.2	64.3	66.33
C (MultiplePCT)	69.4	57.7	59.0	61.50

Table 4.2: Comparison of pose estimation accuracy (compare to [SKN10, p. 6]).

For pose estimation with *soccer* and *basketball* interactions, an increase in accuracy for both methods B and C is visible. Especially for *basketball*, a high increase is achieved. The pose estimation accuracy on the *misc* set is the same for methods A

and B as expected, but drops for method C. This is caused by misclassifications of *misc* interactions. If a *misc* interaction is misclassified, a non-existent interaction object $O(a)$ has been detected near a humans position. The PCT for interaction a will most likely be selected, since other PCTs get penalties for the detected object nearby. The object $O(a)$ "pulls" its PCT towards the false detection location. Thereby the location of many body parts is faulted.

The results achieved for interaction classification are shown in Figure 4.4. Even though examples from the *misc* set have been misclassified as *soccer* only in 10% of the cases, this led to a drop in pose estimation accuracy of 5.3%.

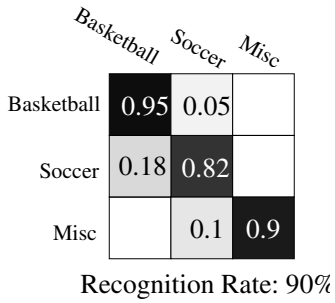


Figure 4.4: Confusion matrix for interaction classification [SKN10, p. 7]. Each row specifies a true class, each column a classification result.

Overall, an improvement in pose estimation accuracy of 9% is achieved for inference on images with a known interaction type, and an improvement of approximately 4% for automatically inferred interaction types. The performance increase is especially high for body parts involved in the interactions (see [SKN10, p. 7]).

4.4.2 Discussion

The discussed approach by Singh et al. shows that pose estimation approaches using Pictorial Structures, e.g. [FH05, Ram07], can be extended to incorporate object interactions without extensive changes. Thereby especially the localization of the body parts participating in the object interaction can be improved.

However, in this approach separate Pose Context Trees are used to model different interactions. In the parsing process, these trees must be matched separately, making the approach infeasible for a large number of modeled interactions. The possibility to use loopy belief propagation [GKD09] to jointly infer over all interactions is mentioned, but not tested by Singh et al. Thus, the approach is well-usable when searching for few, modeled interactions. It must be taken into concern that pose estimation performance might decrease for unmodeled interactions.

For both, object and body part detectors, edge-based HOG features are used to gather information on the shape of objects and are combined with color features to

include information on characteristic appearance features. The iterative combination process used for the body part detectors works well for solving this challenging task.

5 Learning Effective Human Pose Estimation from Inaccurate Annotation

Already during the task analysis it became clear that many training images are needed to capture the enormous variance in the data for HPE in a general scenario. Sam Johnson and Mark Everingham propose an approach to this problem in their work "Learning Effective Human Pose Estimation from Inaccurate Annotation" [JE11]. They tackle HPE in a very challenging setting: the aim is, to realize HPE from 2D color images for very articulated poses as they occur during sports activities.

Johnson and Everingham propose to use Amazon Mechanical Turk[1] (AMT) to acquire a large number of annotated training images and a semi-automatic process for annotation refinement. A new model is needed to deal with the large training dataset. For this purpose, Johnson and Everingham introduce Clustered Pictorial Structure Models. Both, the suggested method for image annotation and annotation refinement, and Clustered PSMs are analyzed in the following sections.

5.1 Motivation and Concept

One of the four analyzed approaches in this thesis is using 3D instead of 2D images (Shotton et al. [SFC+11], analyzed in Chapter 3), but is considered relevant because of its remarkable performance in speed and pose estimation accuracy.

To reach this performance, that approach uses a dataset of 900,000 training images. Thus, it is ensured that the high variation in body shape, clothing, rotation and pose is covered. An originally smaller dataset is extended with synthesized images by retargeting motion capture data on several human models (see Section 3.3.3). This is possible since the depth of human bodies can easily be modeled and rendered for new poses.

It is significantly harder to reach similar performance using regular 2D color images: depth information is not available, which makes foreground/background separation more difficult and the human body and the body parts show additional variance

[1] https://www.mturk.com/mturk/welcome

in color, e.g. in clothing color and in patterns on clothing surface. Taking this into account, an even higher amount of training data is expected to be needed. At the same time, the acquisition of data also gets a lot more intricate in the 2D scenario: the depth shape of the human body can be captured by 3D meshes, the variance in clothing color can hardly be modeled. Thus, generating synthetic examples becomes too difficult.

Sorokin and Forsyth propose to use Amazon Mechanical Turk to get annotations for computer vision problems in [SF08]. AMT is a service platform offering 'Mechanical Turk Requesters' to create tasks (HITs) online and offer a money reward for solving them. Anyone can register as 'AMT worker' and can complete HITs to get the offered rewards.

Sorokin and Forsyth explore the possibilities to use the service to obtain image annotations by offering a small reward on image annotation tasks and let AMT workers do them. They experience a quick and frequent solution of their HITs (see Table 5.1). However, not all of the solutions are usable. Some are faulty or of low quality, because the AMT workers are usually amateur workers and try to solve as many HITs as possible to maximize their payoff.

Experiment	Task	Images	Labels	Cost (USD)	Time	Effective Pay/Hour
1	1	170	510	\$8	750 min	\$0.76
2	2	170	510	\$8	380 min	\$0.77
3	3	305	915	\$14	950 min	\$0.41
4	4	305	915	\$14	150 min	\$1.07
5	4	337	1011	\$15	170 min	\$0.9
Total:		**982**	**3861**	**\$59**		

Table 5.1: Annotation experiments with AMT workers solving image annotation tasks (compare to [SF08, p. 1]). In total, 3861 labels are collected for \$59 payment. A throughput higher than 300 annotations per hour is reached in experiments 4 and 5. Note that the Effective Pay/Hour refers to the effective pay per worker. Multiple workers are involved in each experiment.

Johnson and Everingham decide to use the AMT service nevertheless to get image annotations for Human Pose Estimation. They address the annotation quality problem by using an assisted pre-sort procedure to sort out unusable annotations, and automatic iterative annotation updates for the remaining annotations. The effect of the process is illustrated in Figure 5.1. The method is described in detail in Section 5.3.

Johnson and Everingham apply this method to 10,800 hand-selected images from the Flickr[2] image database. After removing unusable annotations, a dataset of 10,000 images remains. With this very large dataset compared to the current state

[2] http://www.flickr.com/

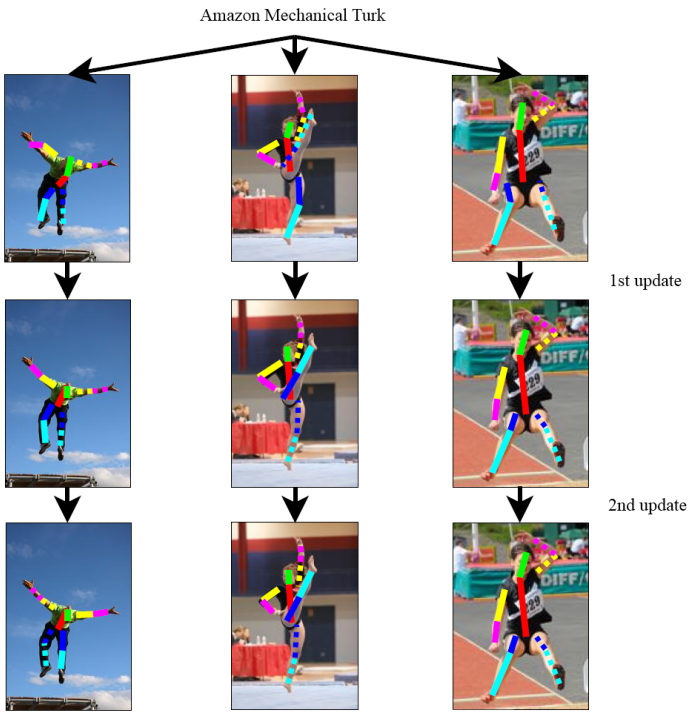
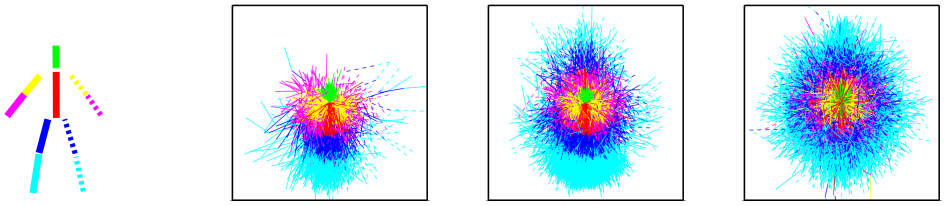


Figure 5.1: Annotation improvements by the semi-automatic update process proposed by Johnson and Everingham (compare to [JE11, p. 1]). Solid and dashed lines represent right and left body parts respectively. **Left Column:** All sides are annotated wrong. This mistake is corrected after the 2nd update. **Middle Column:** The leg sides are corrected after the first update. **Right column:** The annotation positions are improved. This effect is notable across all columns.

of the art, they get one step closer to the high numbers of images needed to capture a lot of the variability in monocular 2D pose estimation.

To give an impression of the improved amount of variance captured with the new dataset, Figure 5.2 shows stick figure plots with the poses of some standard HPE datasets and of the new proposed dataset. With the higher amount of training images, a much better coverage of the pose space is reached.

Learning strategies that can cope with this data with high variance must be used on the new dataset. For this purpose, Johnson and Everingham rely on their approach of Clustered Pictorial Structure Models [JE10] and slightly extend it. They apply clustering twice to reduce the variance in the data for the model learning steps. First, the poses are clustered and one Pictorial Structure Model is learned for each pose cluster. Second, the occurrences of the body parts are clustered within the pose clusters to reduce variability one more time for their visual appearances. This pose estimation method is explained in detail in the following Section 5.2, followed by an explanation of the annotation quality improvement approach in Section 5.3.



(a) Stick figure model. Solid and dashed lines represent right and left body parts.

(b) IIP [Ram07].

(c) LSP [JE10].

(d) The new dataset in [JE11].

Figure 5.2: Stick figure plots, showing the stick figure model (a) and the variation in several datasets (b)-(d) (compare to [JE11, p. 2]).
The stick figure plots show the stick figures in all poses in the respective dataset normalized with respect to the neck position. The new dataset gives a much better coverage of the pose space. **IIP** is short for 'Iterative Image Parsing' and refers to the dataset proposed by Ramanan [Ram07]. **LSP** stands for the 'Leeds Sports Pose' dataset assembled by Johnson and Everingham [JE10].

5.2 Clustered Pictorial Structure Models

Johnson and Everingham model the human body with a Pictorial Structure Model but apply clustering to the poses and to the appearances of the body parts. For a detailed explanation of PSMs, see Section 2.3.4. The most important formulas are shown in the next section to give an introduction to the extension to Clustered PSMs in Section 5.2.2.

5.2.1 The Pictorial Structure Model

Johnson and Everingham model the appearance of a human body as a connected collection of ten body parts (see Figure 2.10): head, torso and upper and lower limbs for each side.

These parts are represented as nodes V in a tree-structured graph (V, E) and connected by the edges E. Each node v_i is connected to an observation node $l_i = (x_i, y_i, \theta_i)$ representing the observed configuration of that body part. The configuration consists of the position and orientation of the part (note that Johnson and Everingham omit the scale, which is used in the original PSM for HPE proposed by Felzenszwalb and Huttenlocher). A complete configuration of the human model is given by the set of all body part configurations $L = \{l_i\}_{v_i \in V}$.

The model is parameterized with the parameter $\Theta = (u, E, c)$, consisting of $u = \{u_1, \ldots, u_{10}\}$ appearance parameters, the set of edges E and the connection parameters c. In this approach, the set of edges E is regarded as given (see Figure 2.10)

and not learned from training data, thus it can be disregarded in the following explanations. The connection parameters $c = \{c_{ij} | (v_i, v_j) \in E\}$ describe the relation between two connected parts. The exact meaning of the connection parameters c_{ij} and the appearance parameters u_i is explained later in this section. Both types of parameters are learned from training data.

With these declarations, the probability for a pose L in image I given the model parameters is formulated (for the derivation, see Section 2.3.4):

$$\begin{aligned} P(L|I,\Theta) &\propto P(I|L,\Theta) \cdot P(L|\Theta) \\ &= \underbrace{\prod_{v_i \in V} P\left(I|l_i, u_i\right)}_{\text{Appearance terms}} \cdot \underbrace{\prod_{(v_i,v_j)\in E} P\left(l_i, l_j|c_{ij}\right)}_{\text{Prior terms}}, \end{aligned} \tag{5.1}$$

with the body part appearance terms $P\left(I|l_i, u_i\right)$ and the prior terms $P\left(l_i, l_j|c_{ij}\right)$ for each pair of connected body parts. In the original publication [JE11], the prior terms are written as $P\left(l_j|l_i, c_{ij}\right)$. This originates from a different (but equivalent) deduction of the the statistical framework. For consistency, the formulation of Felzenszwalb et al. in [FH05] is used throughout this thesis.

Appearance Terms The appearance terms capture the matching probability of body part i at image location l_i and are determined in the approach by Johnson and Everingham using "[...] a mixture of linear SVMs to capture part appearance represented by HOG descriptors" [JE11, p. 4]. This technique is explained in Section 5.2.3. The appearance parameters u_i encode the configurations for these detectors.

Prior Terms The prior terms describe the probability of the configuration l_i, l_j of two connected body parts v_i and v_j. This probability is modeled as Diagonal Gaussian Normal Distribution in a transformed space, similar to Felzenszwalb and Huttenlocher [FH05]:

$$P\left(l_i, l_j|c_{ij}\right) = \mathcal{N}\left(T_{ij}\left(l_i\right) - T_{ji}\left(l_j\right), 0, D_{ij}\right), \tag{5.2}$$

where D_{ij} is a diagonal covariance matrix. Similar to the original approach [FH05], efficient inference on the pose space can be done using Distance Transforms. Johnson and Everingham do not further specify the required transformation functions T_{ij} and T_{ji}, which model the distribution (for a detailed explanation and an example, see Section 2.3.4). It can be assumed that they are defined similar to the

example definition for HPE in Equation 2.25, but omitting the scale:

$$\begin{aligned} T_{ij}(l_i) &= (x'_i, y'_i, \cos(\theta_i + \theta_{ij}), \sin(\theta_i + \theta_{ij})), \\ T_{ji}(l_j) &= \left(x'_j, y'_j, \cos(\theta_j), \sin(\theta_j)\right), \\ D_{ij} &= \operatorname{diag}\left(\sigma_x^2, \sigma_y^2, \frac{1}{k}, \frac{1}{k}\right), \\ P(l_i, l_j | c_{ij}) &\propto \mathcal{N}(T_{ij}(l_i) - T_{ji}(l_j), 0, D_{ij}), \end{aligned} \tag{5.3}$$

where x'_i and y'_i is the position of the connection point (x_{ij}, y_{ij}) for body part i in image coordinates and θ_{ij} being the optimal orientation difference for the two parts. For a detailed explanation of the connection point and the orientation difference, as well as a derivation for the formulas, refer to Section 2.3.4.5.

The positions of the connection points in image coordinates are calculated as

$$\begin{pmatrix} x'_i \\ y'_i \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + R_{\theta_i} \begin{pmatrix} x_{ij} \\ y_{ij} \end{pmatrix}, \tag{5.4}$$

where R_{θ_i} is the matrix for the rotation of angle θ_i around the origin. These steps are made analogue for x'_j and y'_j. With these declarations, the connection parameters c_{ij} are the following

$$c_{ij} = \left(x_{ij}, y_{ij}, x_{ji}, y_{ji}, \sigma_x^2, \sigma_y^2, \theta_{ij}, k\right). \tag{5.5}$$

5.2.2 Extension to Clustered Pictorial Structure Models

For datasets containing a large set of poses, the problem arises that the only parameters in c_{ij} allowing generalization are the standard deviation parameters of the Normal Distribution σ_x^2, σ_y^2 and k. A large set of poses will lead to high standard deviation values, which causes the prior to become uninformative.

Johnson and Everingham address this problem by clustering the images according to the depicted pose. They name the following benefits [JE11, p. 4]:

1. the prior within each cluster is more descriptive by capturing a tighter configuration of parts,
2. dependencies between parts which are not connected in the PSM tree can be captured implicitly,
3. learning part appearance models specific to each cluster captures the correlation between pose and appearance. As an example, the varying appearance of a head from different viewpoints is automatically captured.

A set of PSMs is searched that maximizes the likelihood of the training poses. To find this set, an initial clustering is done using k-means clustering on the training poses normalized with respect to the neck joint.

The k-means algorithm receives k random poses as starting points for cluster centers. All poses are assigned to the cluster center that is closest (the distance measure used is defined in the following paragraph). "For each cluster a PSM prior is then built, and images are re-assigned to the cluster whose model gives the highest likelihood for the corresponding pose. The process is repeated until no further cluster re-assignments occur" [JE11, p. 4].

Johnson and Everingham use an image annotation procedure explicitly excluding invisible or occluded skeletal joints. Thus, a distance measure must be found that can be applied on data with missing joint annotations.

Clustering Incomplete Poses The distance measure is restricted to only take into account skeletal joint annotations which are available for both compared poses.

Let the clusters be $C_1, \ldots, C_k$ with k randomly chosen means $m_1, \ldots, m_k$. The association of poses to clusters is stored in an association matrix with an entry being 1 if the elements of the respective row and column are associated. For the association matrix for the k-means algorithm, this means that the matrix contains an entry $r_{ni} = 1$ if pose vector x_n is assigned to cluster i. The association of poses $x_1, \ldots, x_N$ to means $m_1, \ldots, m_k$ is given by

$$r_{ni} = \begin{cases} 1 & \text{if } i = \underset{j \in [1,k]}{\arg\min}\, d_{mx}(x_n, m_j) \\ 0 & \text{otherwise.} \end{cases} \tag{5.6}$$

The distance function d_{mx} is chosen such that it captures the distance between the joints visible in both of the compared poses, i.e.

$$d_{mx}(m, x) = \sum_{p \in \mathcal{V}^x \cap \mathcal{V}^m} (x_p - m_p)^2, \tag{5.7}$$

where $\mathcal{V}^u$ is the set of visible joints in vector u. The cluster means are also computed with respect to annotation availability.

The modified k-means algorithm for the procedure is as follows:

1. Chose k random poses and set them as initial pose clusters.
2. Learn the connection parameters $\left(\sigma_x^2, \sigma_y^2, \theta_{ij}, k\right)$ for offset and relative orientation from all visible parts in poses assigned to each cluster. Johnson and Everingham do not mention to learn the connection points for each cluster. It can be assumed that these connection parameters are fixed by learning them once from an annotated training set.

3. Reassign each pose to the cluster under which it has the maximum likelihood.
4. If re-assignments occurred in step 3, go to step 2, otherwise finish.

The pose cluster centroids Johnson and Everingham calculated from their dataset are shown in Figure 5.3. Once again, the high diversity in the centroids illustrates the wide range of poses present in the dataset. On the final clusters, all parameters of the PSM are learned, including the appearance parameters.

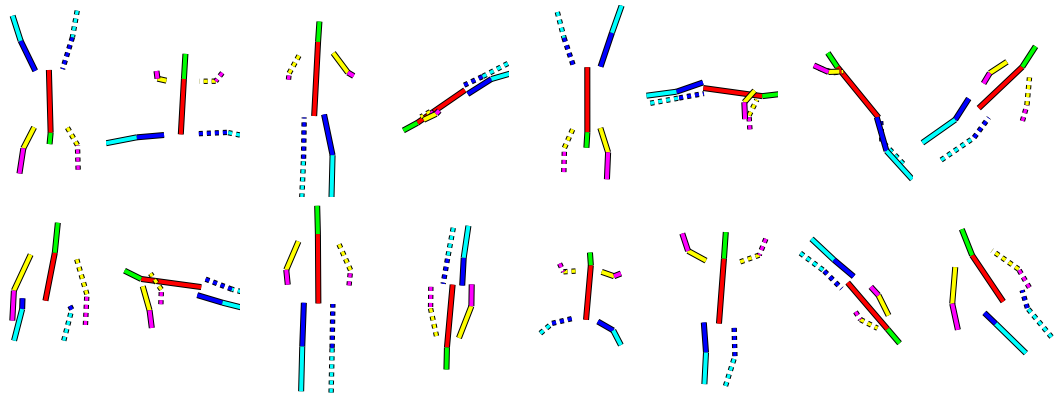


Figure 5.3: Pose cluster centroids calculated on the dataset [JE11] with $k = 16$ clusters [JE11, p. 3].

Pose Cluster Selection To perform pose estimation in an un-labeled, arbitrary image, is is necessary to find the right pose cluster (and thus the right PSM and pose). It is not possible to use the responses of the PSMs directly, since their responses are only proportional to the probability of a pose (see Equation 5.1). This is sufficient to determine the most likely pose within a PSM, but the responses cannot be compared across multiple PSMs.

Johnson and Everingham address this problem by training a multinomial logistic regression classifier [Bis06] on the training data, for which the corresponding pose clusters are known. For the general pose estimation task, they "[...] estimate the pose in each image with all PSMs and select the best using the learnt classifier, i.e. by taking the maximum over weighted PSM outputs" [JE11, p. 5].

5.2.3 Detectors

Even within one of the pose clusters, "[...] the appearance of a body part can be expected to be multimodal, for example due to differences in anatomy, clothing and lighting" [JE11, p. 4]. Johnson and Everingham thus propose to use a mixture of linear SVMs [FRK08] as detector.

This architecture allows to address the multimodal nature of the feature space by again clustering the appearances and training one SVM for each appearance cluster. This architecture allows for efficient computation by using solely linear SVMs and avoiding the use of kernel functions.

Clustering in the Feature Space The part appearance is clustered within each pose cluster "[...] such that an individual linear classifier is responsible for each mode of appearance. Clustering is performed using a weighted form of k-means, with multiple restarts to avoid local minima" [JE11, p. 4]. Local minima can prevent the clustering from reaching a good solution, if the choice of the k starting positions is unlucky.

The distance function of the k-means algorithm is weighted with a specific weight factor w_i for each component i, i.e.

$$d_{mx}(m, x) = \sum_i w_i(x_i - m_i)^2. \tag{5.8}$$

The vector $w = \{w_i\}_i$ is computed as weight vector of one single linear SVM (see Section 2.3.2.2), trained on all the part descriptors for the body part within one pose cluster. The SVM must assign higher weights to components that are vital for the decision than to others. The weights for more descriptive parts of the part descriptor are therefore expected to be higher than the weights for parts mainly laying in the background. Thus, no additional foreground/background annotation or separation step is needed.

Mixture of Linear SVMs For each cluster of part descriptors within a pose cluster, a linear SVM classifier is trained. The number of classifier components per body part is allowed to vary. This enables the approach to capture different levels of variation for different body parts. The amount of components is determined during training by cross-validation (see Section 2.3.3.2). To push detector performance, each "[...] classifier is bootstrapped using negative samples taken from outside the body regions in the training set" [JE11, p. 5].

The response for a part detector can be computed very efficiently. Let $\Phi(f_i)$ be the feature set for an image region f_i, then

$$P(f_i|l_i, \Theta) \propto \max_{j=1,\ldots,n} w_j^T \Phi(f_i), \tag{5.9}$$

where n is the number of mixture components and w_j is the weight vector for mixture component j. The result for each mixture component can be determined by convolving the weight mask w_j with an image of the feature vectors. The final result must be scaled to become a valid probability value.

"The max operator allows the mixture component with the highest confidence to dominate the other components which may be tuned to other appearance modes" [JE11, p. 5]. The part descriptors are chosen as HOG-descriptors (see Section 2.3.1). "These features give a controlled degree of invariance to slight orientation and spatial deformation and are invariant to color which, as noted, is not known a-priori for people present in an image" [JE11, p. 5].

The overall concept for Clustered Pictorial Structure Models is summed up in Figure 5.4. Clustering is applied twice to capture the multimodal nature of the data. For the appearance clusters, the max operator can be used to find the best matching classifier. To determine the best matching PSM, an additional step with Multinomial Logistic Regression is necessary to deal with the unnormalized output from the PSM matching algorithm.

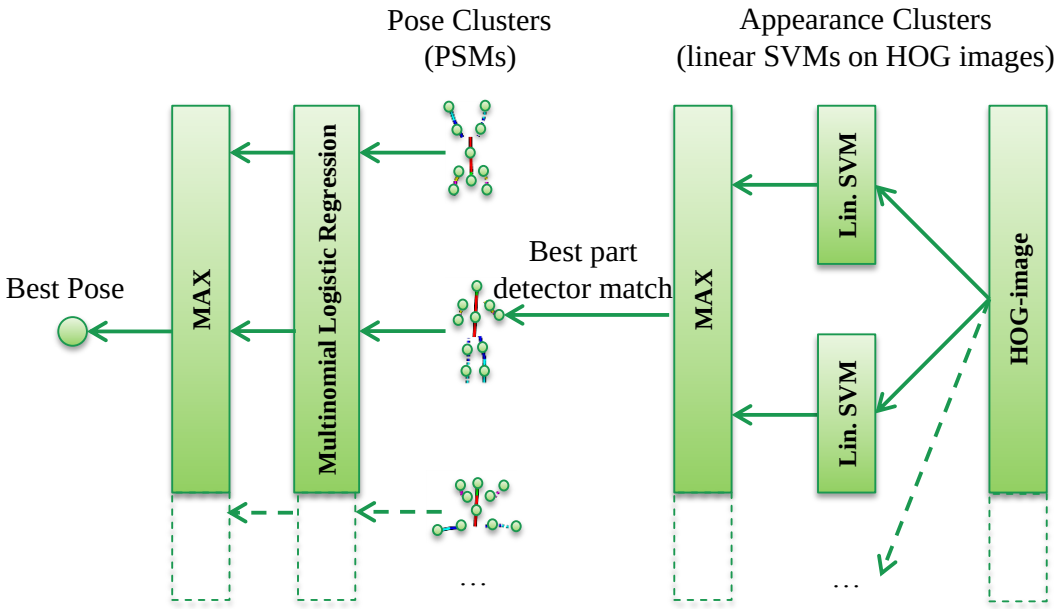
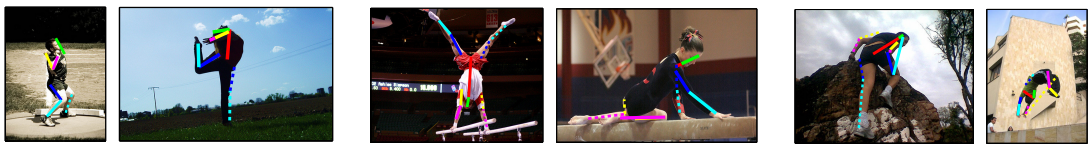


Figure 5.4: Concept illustration for Clustered PSMs. The HOG-image can be calculated in a preprocessing step.

5.3 Learning from Inaccurate Annotations

As explained in Section 5.1, the learning problem for Human Pose Estimation requires many sample images. Johnson and Everingham propose to use Amazon Mechanical Turk to acquire pose annotations quickly and cheaply for many images. However, the annotations can be inaccurate or even unusable. Figure 5.5 shows the three error types for annotated images. Johnson and Everingham propose to model the common structural errors and the annotation inaccuracy and go through an iterative process of learning pose estimation and improving image annotations.



(a) Inaccurate. **(b)** Anatomical structure wrong. **(c)** Unusable.

Figure 5.5: The three error types in annotations received from AMT workers (see [JE11, p. 6]).

5.3.1 Removing Unusable Annotations

In a first step, as many unusable annotations as possible are removed in a semi-automatic process. This preprocessing step is necessary to avoid confusion of the learner in the first iterative learning step (defined in Section 5.3.3).

An important criterion for the process layout when using AMT is that only really unusable annotations are rejected. A rejection of an annotation leaves the AMT worker unpaid.

To create a semi-automatic process, a Gaussian mixture model over part lengths and orientations from a smaller dataset annotated by experts is learned (Johnson and Everingham use the LSP dataset [JE10]). The AMT labeled images are ordered according to their likelihood in the learned model to get a hint which ones might be unusable. The rejections are then selected manually.

5.3.2 Error Modeling

To further improve the annotation quality for the remaining images, the annotation errors are modeled and specifically addressed. The true joint locations are treated as latent variables, "[...] with the AMT annotations being noisy observations thereof" [JE11, p. 5].

Two error models are created, one for location errors and one for structural errors:

1. The joint location annotations are assumed to be distributed according to an isotropic Gaussian Normal Distribution over horizontal and vertical displacement, with the mean at the true location.

2. A set $\mathcal{C}$ with the most common structural error classes is created from a set of sample annotations:
   $\mathcal{C}$ = {No error, Left/right switched, Left/right arms switched, Left/right legs switched, Arms/legs switched, Arms/legs switched and left/right switched, Arms/legs switched and left/right arms switched, Arms/legs switched and left/right legs switched}.
   "Lacking a large amount of data with expert ground truth we assume that the

AMT workers' annotation exhibits each one of these structural errors with equal probability" [JE11, p. 6].

The standard deviation parameter for the Normal Distribution used in model 1 is learned from an annotation sample set of 300 images, for which both, AMT and expert ground truth is available.

5.3.3 Iterative Learning

With these error models, the observed annotations are improved step-by-step. For each structural annotation error c_i in $\mathcal{C}$, the correcting function $c_i(\bar{L}) = L$ switches the respective joint annotations in the pose annotation $\bar{L}$ to the corrected pose annotation L. As an example, for c_i being "Left/right switched", $c_i(\bar{L})$ switches the sides of all limb annotations in $\bar{L}$.

Since $\mathcal{C}$ also contains $c_0 =$ "No error", $c_0(\bar{L}) = \bar{L}$ is the identity function.

With this definition, a bag of possible true locations in an image is defined by $\mathcal{C}(\bar{L})$. This short notation denotes the set of the results of all correcting functions in $\mathcal{C}$ applied on $\bar{L}$.

Let $\bar{L}$ and $\bar{l}_i$ denote the annotated body configuration and location of body part i respectively. The latent, real position of body part l_i is assumed to be equally probable within 2σ of the annotated body part position $\bar{l}_i$. So all considered locations of l_i are in

$$\mathcal{L}(\bar{l}_i) = \{(x, y, \theta) | (x, y, \theta) \text{ is within } 2\sigma \text{ of } \bar{l}_i\}. \tag{5.10}$$

Then, the best possible match of body part l_i given the observed body configuration $\bar{L}$ is given at the position of the best match of the part detector in this area, i.e.

$$l_i^*(\bar{L}) = \arg\max_{l_i \in \mathcal{L}(\bar{l}_i)} P(f_i | l_i, \Theta), \tag{5.11}$$

where $P(f_i|l_i, \Theta)$ is calculated as specified in Equation 5.9. The most likely real pose L^* given the observed pose annotation $\bar{L}$ can than be obtained by finding the most probable pose after applying all possible corrections, i.e.

$$L^* = \arg\max_{L \in \mathcal{C}(\bar{L})} \prod_{v_i \in V} P(I | l_i^*(L), \Theta). \tag{5.12}$$

The pose annotation is updated with this improved estimate in a two-step process. The algorithm is defined as:

1. learn an initial set of part appearance models from the raw annotations,
2. update the observed joint locations and body structure with respect to the learned error models.

This process is repeated a few times: "Iterations alternating between update of the model and annotation can continue until the annotation does not change. In practice [...] we find that just a few iterations are sufficient" [JE11, p. 6].

5.4 Results

5.4.1 Evaluation

Johnson and Everingham aim to "[...] improve performance primarily on the more challenging poses humans take" [JE11, p. 5]. Thus, they select images from the Flickr image database tagged with "gymnastics", "parcour" and "athletics" and create a dataset of 10,800 images and use AMT to obtain annotations. Removing unusable annotations reduces the dataset size to 10,000 images.

The learning and pose estimation approach is evaluated with this new dataset combined with the LSP dataset [JE10]. For details on the evaluation protocol, see [JE11, p. 6].

5.4.1.1 Optimal Clustering

The presented dataset contains highly varying poses, as illustrated in Figure 5.2. This fact hints at the necessity of a high amount of pose clusters. The proposed learning approach does not find the optimal amount of pose clusters automatically.

Table 5.2 shows part localization accuracy as function of the amount of pose clusters used. The number of pose clusters used for pose estimation is determined experimentally by training with different numbers and choosing the number with the best performance.

Pose Clusters	1	2	4	8	16	32
Detector Performance	34.7%	49.5%	53.4%	56.0%	**58.5%**	57.9%

Table 5.2: Body part localization accuracy as function of the number of pose clusters, [JE11, p. 6]. 16 pose clusters provide the best results on the used dataset.

The pose space can be partitioned into up to 16 clusters with increasing performance for this dataset size. At higher numbers, overfitting is observed. The increasing detection accuracy with higher amounts of pose clusters shows "[...] that pose estimation benefits from a larger amount of increasingly varied training data" [JE11, p. 7].

5.4.1.2 Annotation Updates

The annotation update procedure is evaluated by training on the joint dataset of the AMT annotated images (10,000) and the LSP training dataset (500). It is evaluated on the remaining 500 images of the LSP dataset. Table 5.3a shows the achieved pose estimation accuracy.

(a) Part localization rates in percent on the LSP test set (compare to [JE11, p. 7]).

Method	Total	Torso	Upper Leg	Lower Leg	Upper Arm	Forearm	Head
Clustered PSM [JE10] (1,000 training images)	55.1	78.1	64.8	66.7	60.3	57.3	48.3	46.5	34.5	31.2	62.9
Proposed (16 clusters, no updates)	59.9	85.9	71.4	71.6	63.4	63.8	50.5	52.1	33.9	35.6	70.8
Proposed (16 clusters, 1 update)	62.1	87.0	74.6	73.0	66.3	65.0	52.9	54.1	**37.1**	37.5	73.6
Proposed (16 clusters, 2 updates)	**62.7**	**88.1**	**75.2**	**73.8**	**66.7**	**66.3**	**53.0**	**54.4**	36.1	**38.9**	**74.6**

(b) Part localization rates in percent on the IIP test set [Ram07] compared with other methods (see [JE11, p. 8]). Wang and Kollers method is analyzed in the following Chapter 6.

Method	Total	Torso	Upper Leg	Lower Leg	Upper Arm	Forearm	Head
Ramanan [Ram07]	27.2	52.1	30.2	31.7	27.8	30.2	17.0	18.0	14.6	12.6	37.5
Andriluka et al. [ARS09]	55.2	81.4	67.3	59.0	63.9	46.3	47.3	47.8	31.2	32.1	75.6
Wang and Koller [WK11]	61.51	88.3	71.2	**77.6**	**73.7**	56.1	48.8	51.7	36.1	36.6	75.1
Johnson and Everingham [JE10]	66.2	85.4	**76.1**	10.7	69.8	61.0	64.9	64.4	**49.3**	**44.4**	76.1
Proposed method	**67.4**	**87.6**	**76.1**	73.2	68.8	**65.4**	**69.8**	**64.9**	48.3	43.4	**76.8**

Table 5.3: Body part localization accuracy values.

After one update, the overall accuracy increases by 2.2 percentage points, with increased accuracy for all body parts. This "[...] shows that our update scheme successfully aligns the data leading to stronger appearance models" [JE11, p. 7]. See Figure 5.6 for three examples of improved annotations after one update. After only one iteration, all three pose annotations are improved considerably.

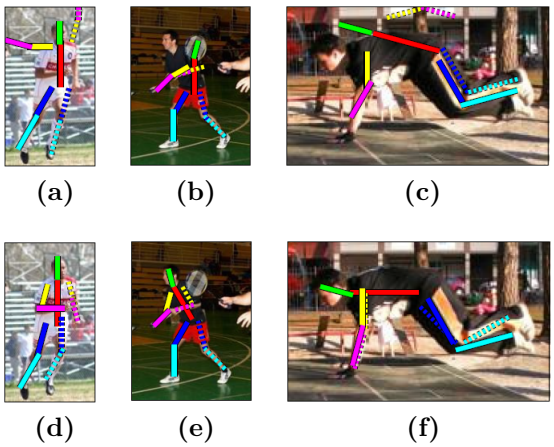


Figure 5.6: Annotation update results (compare to [JE11, p. 7]). **Top row:** original annotations. **Bottom row:** improved annotations after one update.

5.4.1.3 Final Results

With one more round of updates, still a slight improvement can be reached for nearly all body parts (see Table 5.3a). However, the performance for the left forearm decreases. In comparison to the approach described in [JE10], an improvement of 7.6 percentage points is reached. The improvement is very high for some body parts, e.g. the increase of 11.7 percentage points for the head.

Example output of the final pose estimation system is shown in Figure 5.7. It is able to handle very articulated poses like the ones in subfigures (b) and (h) and images with very noisy backgrounds like in images (g) and (i).

The system is evaluated on the IIP dataset [Ram07] and the results are given in Table 5.3b. Only a modest improvement compared to the previous approach [JE10] is achieved: "As noted, this dataset [the IIP test set] is small, consisting mainly of upright poses, such that the improvement in 'harder' poses is not more apparent" [JE11, p. 8].

5.4.2 Discussion

The work by Johnson and Everingham [JE11] shows twice (once for the poses, once for the appearances) how multi-modal spaces can be handled by clustering.

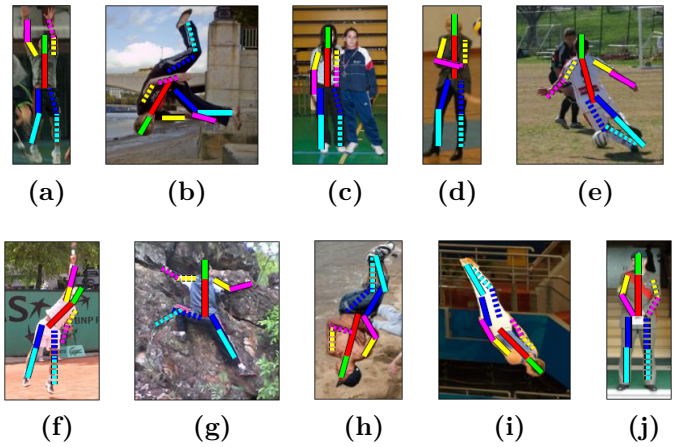


Figure 5.7: Example pose estimation results (compare to [JE11, p. 7]).

The approach shows high scalability and can be expected to scale with even higher numbers of training images.

At the same time, a method must be found to determine the right cluster when using classifiers with unnormalized outputs. The approach presented in [JE11] applies all clustered Pictorial Structure Models to select the most probable one with a multinomial regression model. However, this is connected to increasing computational cost with more clusters. The mixture of linear SVMs is presented as efficient and capable classifier for clustered HOG appearance features, keeping the computational cost low for a small number of pose clusters.

The number of pose clusters can not yet be determined automatically. Johnson and Everingham select it by trying several numbers and hand-selecting the best one. Also, the distance measure for the k-means clustering is problematic, since poses can get low distance values in pose space that have few annotated joints in common. This might lead to difficulties and could be corrected by incorporating the amount of common annotated joints into the distance measure.

Amazon Mechanical Turk is shown to be an efficient way to get pose annotations for many images. The presented updating method can correct many of the most common mistakes and improve the annotation quality. However, it is also possible that it leads to decreasing classifier accuracy if it is used for too many iterations, as discussed in Section 5.4.1.2.

6 Multi-Level Inference by Relaxed Dual Decomposition for Human Pose Segmentation

An image has several levels of information: the lowest level is the pixel level with a measurement information for each pixel. Depending on the applied interpretation, additional levels arise from this low level of information. One of the higher levels can contain semantic information about depicted objects and people.

The task of foreground/background segmentation works on the lowest level, the pixel level. As already shown in Chapter 3, it can be an important provider of information for HPE, reducing the complexity for this task. Pose estimation is a problem dealing with body parts - and thus works on a higher semantic level. Huayan Wang and Daphne Koller propose in their publication "Multi-Level Inference by Relaxed Dual Decomposition for Human Pose Segmentation" [WK11] an approach combining the solution processes of both problems to create mutual benefits. Their approach is analyzed in this chapter.

6.1 Motivation and Concept

The foreground/background segmentation can be an important provider of information for the HPE task. The approach by Shotton et al. explained in Chapter 3 can classify a pixel to one of the body parts just with segmentation information with 0.465 mAP (mean average precision). The other way round, HPE can help to achieve foreground/background segmentation by providing information on where body parts are located.

Wang and Koller try to use these synergies to build an approach on top of optimization strategies for both levels: on the one hand, a PSM for Human Pose Estimation (see Section 2.3.4) is used to efficiently find optimal solutions on the higher semantic level. On the other hand, several energy functions are used to specify a selection of pixels that are likely to lie in the foreground. These energy functions are optimized with a graph-cut algorithm (an efficient energy minimization technique [KZ04, BJ01]).

To be able to do inference for both tasks simultaneously, a unified model is created. Energy terms concerning both levels are part of this model. This multi-level

model with intractable energy terms is treated using a relaxed variation of Dual Decomposition (an Operations Research technique): "[...] the inference problems on different levels are solved independently as slave problems but communicate via coordinating messages in the dual decomposition framework. The intractable energy terms are handled by further relaxation on the dual function" [WK11, p. 2]. This results in the dual solutions: pairs of solution proposals for the higher level (HPE) and for the pixel level (segmentation).

The best primal solution (valid overall solution, consisting of an estimated pose and a foreground/background segmentation) is determined by evaluating the original energy function on "[...] a handful of solutions [...]" [WK11, p. 2] constructed from the dual solutions and selecting the one with the lowest energy.

Figure 6.1 shows the segmentation and pose estimation results for the first and the final state in the optimization process of the proposed method. Since the approach combines pose estimation and foreground/background segmentation, Wang and Koller call it "Human Pose Segmentation".

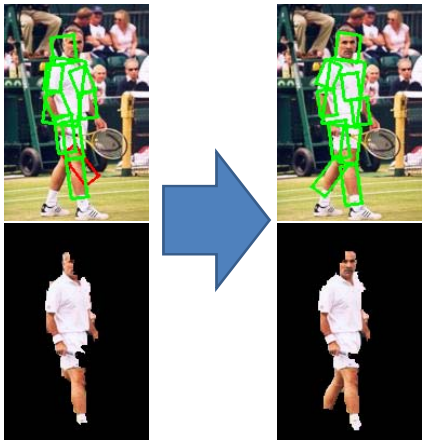


Figure 6.1: Example for simultaneous inference for pose estimation (**top row**) on a semantic level and (**bottom row**) on pixel level [WK11, p. 1]. The left image pair shows the results for the first iteration, the right image pair the final results of the optimization process.

6.2 Model Formulation

The model consists of two parts: a graph with nodes and edges, and an energy function based on this graph. They are explained in the following two sections. Note that some identifiers of variables are written in superscript form.

6.2.1 The Graphical Model and Variables

The graphical model provides the variables for the energy function. Since the energy function is designed to capture variables from two semantic levels (pixels and body parts), the graph also represents variables from both levels. An illustration of the graph is given in Figure 6.2.

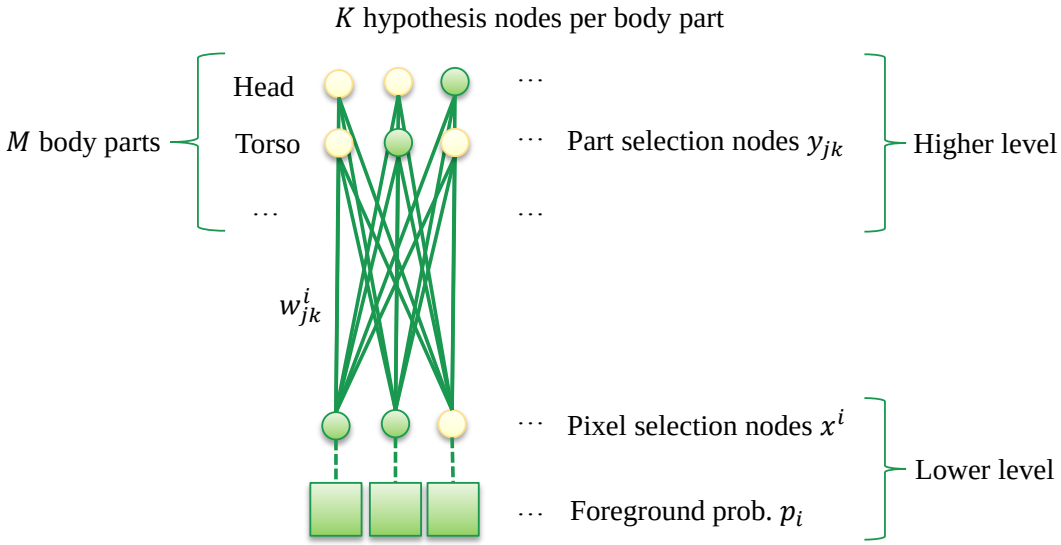


Figure 6.2: Illustration of the graphical model constructed by Wang and Koller. The **round nodes** represent binary selection variables, whereas the **squares** represent probabilities. **Solid lines** show edges and the **dashed lines** visualize the association of the foreground probability with the respective pixel.

Body Part Selection Nodes The appearance of the human body is modeled as a configuration of ten body parts: head, torso and upper and lower limbs (see Figure 2.10). For each of the body parts, the k best hypotheses are obtained from a preprocessing step. It can be treated "[...] as a black box to subsequent steps" [WK11, p. 2].

Wang and Koller obtain the hypotheses by using a HPE approach of Andriluka et al. [ARS09] (a Pictorial Structure Model with HOG/SVM based detectors for Human Pose Estimation). They consist of position and orientation for the respective body part. Each hypothesis is represented by one body part selection node in the graph.

These nodes in the graph refer to the variables $\{y_{jk}\}_{j\in[1;M],k\in[1;K]}$ for M body parts and K hypotheses per body part. The index j is the index of the body part, the index k is the index of a hypothesis for that part. The entire selection of body parts is represented by the vector $y = (y_{jk})_{j\in[1;M],k\in[1;K]}$, with $y \in \{0;1\}^{M\cdot K\times 1}$. All variables y_{jk} are binary, being one if the hypothesis is selected and zero otherwise. With $M = 10$ body parts and $K = 10$ top hypotheses per body part, Wang and

Koller use 100 nodes in the graphical model and 100 variables in the energy function to represent the body part hypotheses.

For a valid selection, exactly one of the hypotheses for each body part must be selected. This can be expressed with the constraint:

$$\forall j : \sum_{k=1}^{K} y_{jk} = 1. \tag{6.1}$$

The set of body part hypothesis selections for which this constraint is fulfilled is called the feasible set $\mathcal{C}$. Any valid selection of hypotheses y must be in $\mathcal{C}$.

Edges Edges in the graphical model only exist between body part selection nodes and pixel nodes. They capture the connection in probability of a pixel laying in the foreground and a body part hypothesis detection being close to its location.

Each body part hypothesis node y_{jk} provides a position and orientation. A Gaussian mask is computed and assigns a weight to edges going from part hypothesis node y_{jk} to all pixel nodes. To reduce the total amount of edges, an edge is removed if the weight for its pixel is smaller than a specified threshold (chosen by Wang and Koller as 10^{-4}). The edge weight equals one for pixels laying right on the skeleton and decreases quickly for pixels further away from it.

The edge weights are represented by the variables $\{w_{jk}^i\}_{i\in[1;N],j\in[1;M],k\in[1;K]}$ (notation: the weight of the edge from body part hypothesis node y_{jk} to pixel node i). Since the weights are computed as values of the Gaussian mask, their values are between the threshold value (min) and one (max).

Pixel Selection Nodes The pixel selection nodes represent the foreground/background segmentation. Every image pixel is represented by one node in the graphical model. The entire image segmentation is denoted by the vector $x = (x^i)_{i\in[1;N]}$ for an image with N pixels, with x^i being the selection node and variable for the pixel with index i. All variables x^i are binary, with one encoding foreground and zero background assignment, thus $x \in \{0;1\}^{N\times 1}$.

Foreground Probability The appearance of the foreground and background can vary strongly between images in a general scenario. For this reason the according probabilities for pixels are estimated with a model that is learned specifically for each analyzed image. The aim is, to get a smooth probability model for foreground/background regions.

Therefore a Gaussian Mixture Model (see [Bis06, pp. 430–439]) is fitted to the pixel weights. The pixel weights are defined as the sum of the weights of the pixels adjacent edges, i.e. for pixel i: $\sum_{j,k} w_{jk}^i$. Similarly, a background model is created using $\left(1 - w_{jk}^i\right)$ edge weights, i.e. for pixel i: $\sum_{j,k}\left(1 - w_{jk}^i\right)$.

The Gaussian Mixture Model approximates a density function by estimating the means and standard deviation parameters of several Normal Distributions. The value of the Gaussian Mixture Model at one pixel is the sum of the weighted distributions at the pixel position. Thus, the model provides a smooth probability estimate for the foreground probabilities.

Wang and Koller name no details on the training of the Gaussian Mixture Model. They also do not mention a normalization step that must be necessary, since e.g. the pixel weight $\sum_{j,k} w_{jk}^i$ for the foreground model can become greater than one. The estimated foreground pixel probability for each pixel i is denoted as $\{p_i\}_{i\in[1;N]}$. It is computed once and the results are stored for the rest of the inference procedure.

With the calculation of the foreground probabilities, the first part of the process is finished. "Note that w_{jk}^i and p_i are pre-computed and fixed during the entire inference procedure" [WK11, p. 3], i.e. the model nodes, the weights and the edges and also the pixel foreground probabilities don't change during the iterative optimization steps. In the following section, the energy function that is used for the optimization is defined.

6.2.2 The Energy Function

The definition of the energy function is crucial for the results of the algorithm, since it specifies the criteria on how an optimal combination of nodes in the graphical model is determined. To get an intuition for the use of the energy function, it can be seen as specification for the amount of energy that is needed to use a solution as final result. The minimization process tries to find a solution with an energy value as low as possible.

For the structure of the energy function this means that "bad configurations" must be penalized with high energy values and preferable ones should be able to reach low energy values.

Wang and Koller design the energy function as linear combination of several energy terms weighted with the factors γ_l, $l \in [1;5]$. It is denoted as primal energy function E^P. The minimization problem is defined by Wang and Koller as:

$$\begin{aligned}\min_{x,y} E^P(x,y) \quad = \min_{x,y}[\quad & \gamma_1 E^{\text{pixel-simple}}(x) + \gamma_2 E^{\text{pixel-pair}}(x) + \\ & \gamma_3 E^{\text{part-pixel}}(x,y) + \\ & \gamma_4 E^{\text{part-single}}(y) + \gamma_5 E^{\text{joints}}(y) + \\ & E^{\text{residual}}(x,y)\,], \\ & \text{subject to } y \in \mathcal{C}. \end{aligned} \tag{6.2}$$

The part energy terms are carefully designed to have the aforementioned properties. Notice, that the ones in the first line only depend on x, the ones in the third line

only depend on y, and E^{residual} and $E^{\text{part-pixel}}$ depend on both variables. Each of them is explained in the following paragraphs. In these explanations, $\mathbf{1}(f)$ is the indicator-function, being 1 if $f \vDash true$ and 0 otherwise.

$E^{\textbf{pixel-single}}(x)$ This term measures the quality of a pixel selection in regard to the probabilities for each pixel to lie in the foreground: a selection needs more energy the more pixels are selected as foreground pixels with low foreground probabilities. It therefore restricts the foreground selection to pixels with high foreground probability:

$$E^{\text{pixel-single}}(x) = \sum_{i=1}^{N} \underbrace{\mathbf{1}\left(x^i = 1\right)}_{\text{If pixel selected}} \cdot \underbrace{\log \frac{1-p_i}{p_i}}_{\text{Energy contribution}}. \tag{6.3}$$

The function used to calculate the energy contribution to the sum ($\log \frac{1-p_i}{p_i}$) has high positive values for low p_i probabilities and high negative values for high p_i values (see Figure 6.3). It is undefined for $p_i = 0$ and $p_i = 1$, but these values do not occur since the probability values are estimated with the Gaussian Mixture Model. However, this must be kept in mind to avoid rounding issues.

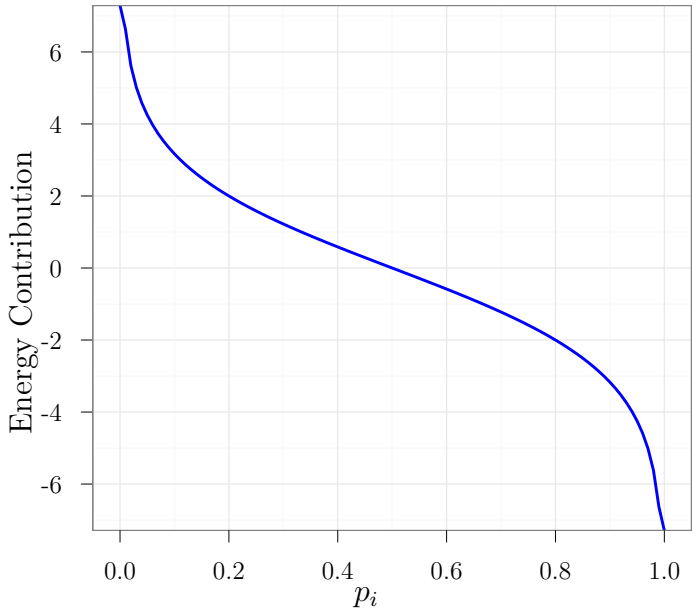


Figure 6.3: Function used to calculate the energy contribution of a pixel with foreground probability p_i in the foreground.

$E^{\textbf{pixel-pair}}(x)$ The pixel-pair-term deals with the contrast between the foreground and background pixel set. A higher contrast between the pixels of the two

sets is preferred and a low contrast penalized. The term is specified as:

$$E^{\text{pixel-pair}}(x) = \sum_{i,j} \underbrace{\mathbf{1}\left(x^i \neq x^j\right)}_{\text{Selected and deselected}} \cdot \underbrace{e^{\left(-\theta_\beta \|I_i - I_j\|^2\right)}}_{\text{Energy contribution}}, \tag{6.4}$$

with $\|.\|$ being the Euclidean Norm and I_a being the intensity value for the pixel with index a. The indicator function for this term selects each pair of pixels with one pixel laying in the foreground and one in the background. The energy contribution is specified with an exp-function with negative exponent and a constant factor θ_β. The larger the contrast between the pixels (larger difference in intensity), the lower the energy contribution.
The factor θ_β is calculated as $\frac{1}{2\left\langle \|I_i - I_j\|^2 \right\rangle_{i,j}}$, with $\langle . \rangle_{i,j}$ denoting the average over all combinations of pixels. For details on the definition of this function, see [BJ01].
Note that the indicator function "selects" each pair of accepted pixels twice. It is probably not intentionally formulated this way by Wang and Koller and implemented differently for the sake of efficiency.

$E^{\textbf{part-pixel}}(x, y)$ This energy term is the first one creating a link between the two semantic levels. It relies on the edge weights to determine the energy contribution.

$$E^{\text{part-pixel}}(x, y) = \sum_{i,j,k} \underbrace{\mathbf{1}\left(x^i = 0, y_{jk} = 1\right)}_{\text{Pixel in background, part selected}} \cdot w^i_{jk}. \tag{6.5}$$

It sums up the weights of the edges connecting pixels in the background to selected body parts. This way, it enforces pixels to be selected as foreground that are close to selected body parts.

$E^{\textbf{part-single}}(y)$ The part-single energy term depends on the selection of body part hypotheses, the weight of their adjacent edges and the foreground probabilities of connected pixel selection nodes. It contributes energy as the weighted probabilities to be in the background for associated pixels to selected hypotheses. In this way it supports the selection of part hypotheses that agree with the learned appearance model. The mathematical formulation is:

$$E^{\text{part-single}}(y) = \sum_{j,k} \underbrace{\mathbf{1}\left(y_{jk} = 1\right)}_{\text{Hypothesis selected}} \cdot \underbrace{\sum_i w^i_{jk}\left(1 - p_i\right)}_{\text{Sum of weighted prob. to lay in the background}}. \tag{6.6}$$

$E^{\textbf{joints}}(y)$ The energy contribution of this energy function depends on the overall configuration of the body model. It behaves opposite to the likeliness of the configuration of the selected body parts. This likeliness is modeled by Wang and Koller in the same way as it is done by Andriluka et al. [ARS09] using PSMs for Human Pose Estimation. An example definition can be found

in Section 2.3.4, with the definition of the configuration prior probability. Wang and Koller do not specify how exactly they define the energy contribution. A possible choice is the sum of distance functions, the second part of Equation 2.22.

$E^{\textbf{residual}}(x, y)$ This last energy function is used to avoid unlikely, but possible configurations regarding the link between the two levels. It is itself the sum of two energy functions: $E^{\text{residual}}(x, y) = E^{\text{r1}}(y) + \gamma_6 E^{\text{r2}}(x, y)$.
The first energy function contributes high energy values for pixels likely to lie in the foreground, but not being close to a selected body part. The indicator function for these pixels has the form

$$\iota(i, \delta) = \mathbf{1}\left(\max_{j,k}\left(y_{jk} w^i_{jk}\right) < \delta\right), \tag{6.7}$$

with δ as threshold. This indicator function is 1 if the maximum edge weight to a selected body part node for this pixel is below the threshold δ, i.e. if no selected body part hypothesis in the image is close. $E^{\text{r1}}(y)$ is defined as:

$$E^{\text{r1}}(y) = \frac{\sum_i \iota(i, \delta_1) \cdot \log \frac{p_i}{1-p_i}}{\sum_i \iota(i, \delta_1)}. \tag{6.8}$$

The occurring log function works similar to the one explained for $E^{\text{pixel-single}}$, but with high values for high foreground probabilities and vice versa.

This means that for every pixel with no selected body part hypothesis close, a high energy contribution is made if it has a high probability to be in the foreground (independent of whether the pixel is selected or not). The term is normalized with the total number of pixels with no selected body part hypothesis close, assigning a higher energy for fewer unexplained pixels. This penalizes few "unexplained" pixels especially strong.

With this strategy, Wang and Koller avoid overlapping selections of e.g. the leg hypotheses. For an overlapping selection of leg hypotheses with another non-overlapping leg hypothesis available, the pixels of the other hypothesis also have high foreground probability. They are further away from any selected body part (and have higher energy contribution) for the overlapping selection.

$E^{\text{r2}}(x, y)$ penalizes configurations with pixels far from selected body parts labeled as foreground. It uses a similar indicator function as $E^{\text{r1}}(y)$, but with a different threshold δ_2:

$$E^{\text{r2}}(x, y) = \sum_i \underbrace{\iota(i, \delta_2)}_{\text{Not strongly linked to a selected body part}} \cdot \underbrace{x^i}_{\text{Pixel selected}}. \tag{6.9}$$

Wang and Koller state, that they "[...] use a much smaller value for δ_2 than for δ_1" [WK11, p. 4]. This strategy is necessary, since a lower value for δ means that only pixels are selected by the indicator function that lay further away from selected body parts. Whereas $E^{\mathrm{r1}}(y)$ should also consider pixels in closer regions to the body ($\Rightarrow$ higher δ_1 value), $E^{\mathrm{r2}}(x, y)$ specifically handles pixels further away ($\Rightarrow$ lower δ_2 value), so the values for δ must be chosen accordingly.

6.3 Energy Optimization using Dual Decomposition

With the composition of all energy terms described in preceding section, Wang and Koller define a minimization problem in Equation 6.2. The problem works on two semantic levels, as explained in Section 6.2.1:

- The higher level contains the body part selection nodes. It connects them and is modeled as PSM for Human Pose Estimation (see Section 2.3.4). Thus, the best selection for y can be determined with well-known, efficient solution strategies.
- The lower level contains pixel potentials. An optimal selection can be found using the graph-cut algorithm ([KZ04, BJ01]).

However, it is not clear how a joint solution can be found with respect to the linking energy functions between the two levels. Wang and Koller solve this problem by splitting the joint problem into two part problems within the two levels and approximate a global solution with additional processing steps. The first aim is to split the problem accordingly.

6.3.1 Formulation of the Lagrange Dual Problem

To be able to split the problem into subproblems, the body part selection variable y is copied. The two instances of this variable are denoted by y^0 and y^1. Both instances must have the same values, so an additional constraint is added: $y^0 = y^1$. Thus, it is sufficient to only check one of the copies for consistency with the feasibility constraint.

This leads to the following optimization problem, which is equivalent to the one specified in Equation 6.2:

$$\begin{aligned}\min_{x,y^0,y^1}[\ & \gamma_1 E^{\text{pixel-simple}}(x) + \gamma_2 E^{\text{pixel-pair}}(x) + \\ & \gamma_3 E^{\text{part-pixel}}(x, y^0) + \\ & \gamma_4 E^{\text{part-single}}(y^1) + \gamma_5 E^{\text{joints}}(y^1) + \\ & E^{\text{residual}}(x, y^1)\ ], \\ & \text{subject to } y^0 = y^1,\ y^1 \in \mathcal{C}. \end{aligned} \tag{6.10}$$

The energy terms for the two semantic levels are captured by

$$E^0\left(x, y^0\right) = \gamma_1 E^{\text{pixel-simple}}(x) + \gamma_2 E^{\text{pixel-pair}}(x) + \gamma_3 E^{\text{part-pixel}}(x, y^0), \quad (6.11)$$

$$E^1\left(x, y^1\right) = \gamma_4 E^{\text{part-single}}(y^1) + \gamma_5 E^{\text{joints}}(y^1). \quad (6.12)$$

With the two terms E^0 and E^1 and the connecting residual term, the minimization problem can be written as

$$\min_{x, y^0, y^1} \left[E^0\left(x, y^0\right) + E^1\left(y^1\right) + E^{\text{residual}}\left(x, y^1\right)\right], \quad \text{subject to } y^0 = y^1,\ y^1 \in \mathcal{C}. \quad (6.13)$$

Lagrange Dual Function The next step is to integrate the new equality constraint into the minimization problem by using Lagrange Multipliers. The constraint can be formulated as $y^0 - y^1 = 0$. The term $\lambda \cdot (y^0 - y^1)$ is added to the energy function, leaving the factor λ as parameter. The parameter specifies the "penalty" for each violation of the constraint. This leads to the formulation of the Lagrangian and the Lagrange dual function $g(\lambda)$:

$$g(\lambda) = \min_{x, y^0, y^1} \overbrace{\left[E^0\left(x, y^0\right) + E^1\left(y^1\right) + E^{\text{residual}}\left(x, y^1\right) + \lambda \cdot \left(y^0 - y^1\right)\right]}^{\text{Lagrangian}}, \quad \text{subject to } y^1 \in \mathcal{C}, \quad (6.14)$$

with $\lambda \in \mathcal{R}^{1 \times J \cdot K}$ and $y^0,\ y^1 \in \{0; 1\}^{J \cdot K \times 1}$.

The Lagrange dual function gives for each value of λ a lower bound for the optimal value of the problem. For a proof of this important property, see [BV04, p. 216]. The proof can be sketched in a few lines: let p^* be the optimal value of the primal problem, i.e.

$$p^* = \min_{x, y} E^{\text{P}}(x, y). \quad (6.15)$$

Let $(\tilde{x}, \tilde{y}^0, \tilde{y}^1)$ be a feasible solution for the Lagrange dual function, i.e. $\tilde{y}^1 \in \mathcal{C}$ and the equality constraint $\tilde{y}^0 = \tilde{y}^1$ holds. In this case, independent of the value of λ, the value of $g(\lambda) = p^*$, since $\tilde{y}^0 - \tilde{y}^1 = 0$ and the problem reduces to the primal problem.

However, the Lagrangian is minimized instead of the primal function, allowing violations of the constraints. This causes the Dual function to reach lower energy values at the price of differing values for y^0 and y^1. These findings are summed up in the inequality

$$g(\lambda) \leq p^*. \quad (6.16)$$

Lagrange Dual Problem In the case of the energy function defined by Wang and Koller, $g(\lambda)$ is a tight lower bound for the primal problem (see [WK11, p. 5]). This means that for some λ the value of $g(\lambda) = p^*$. To find this solution, the Lagrange dual function must be maximized with respect to λ to find its maximum value d^*, and for a tight lower bound $d^* = p^*$. The Lagrange dual problem is given by determining

$$d^* = \max_{\lambda} g(\lambda). \tag{6.17}$$

A visualization of a simple Lagrange dual problem can be found in Figure 6.4. The plot shows the function values of a Lagrange dual function $g(\lambda)$ and the optimal value p^* for a minimization problem. In this example, no feasible solution can be reached by the dual function, i.e. the dual function is not a tight lower bound for the problem. However, a close approximation for p^* can be determined.

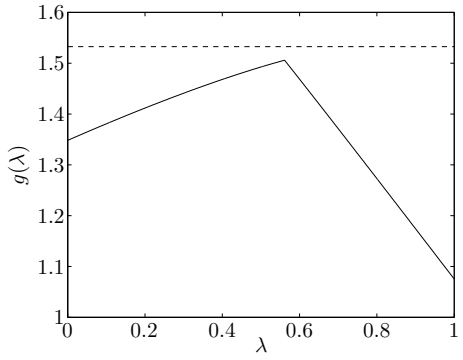


Figure 6.4: Visualization of a Lagrange dual problem. The dual function $g(\lambda)$ gives for each value of λ a lower bound for the optimal value p^* of the primal minimization problem. The value p^* is illustrated by a dashed line.

6.3.2 Relaxation of the Lagrange Dual Problem

The values of the minimization of the part problems can still not be determined by applying the efficient optimization strategies. Therefore, some of the constraints are relaxed to give a relaxed dual function $\tilde{g}(\lambda)$. This is done by decomposing the minimization problem in the Lagrange dual function into three minimization problems:

$$\begin{aligned} \tilde{g}(\lambda) \quad = \quad & \min_{x,y^0} \left[E^0\left(x, y^0\right) + \lambda \cdot y^0 \right] + \\ & \min_{y^1 \in \mathcal{C}} \left[E^1\left(y^1\right) - \lambda \cdot y^1 \right] + \\ & \min_{\bar{x},\bar{y} \in \mathcal{C}} \left[E^{\text{residual}}(\bar{x}, \bar{y}) \right]. \end{aligned} \tag{6.18}$$

Splitting the minimization problem for the two part problems E^0 and E^1 can be achieved without any approximations, because the part problems do not have any variables in common. However, the residual energy term E^{residual} with dependencies on the solutions of both part problems cannot be integrated. In an approximation step, it is left out of the minimization of the two other part problems and two new variables $\bar{x}$ and $\bar{y}$ are introduced. They have the same dimensionalities of the variables x and y respectively. The calculation of $\bar{x}$ and $\bar{y}$ will be explained in Section 6.3.3.

With $\tilde{g}(\lambda)$, a relaxed version of the Lagrange dual problem is defined. Since constraints were omitted during the approximation, the inequality $\tilde{g}(\lambda) \leq g(\lambda)$ holds. Unfortunately, the relaxed version $\tilde{g}(\lambda)$ is not a tight lower bound for the optimization problem any more. Maximizing it can approximate good solutions, but p^* can not be reached in general. The value of $\tilde{g}(\lambda)$ is maximized to find as good approximations as possible. Wang and Koller use a gradient ascent method for this purpose.

Dual Solution Candidates Gradient ascent works by determining the gradient of the function and moving from the current λ value to a new λ value in the gradients direction. Since the residual term is a constant when differentiating with respect to λ, it does not influence this optimization step.

Mathematically formulated, the gradient ascent method updates the value of λ iteratively in the direction of the gradient $\nabla\tilde{g}(\lambda)$, scaled with a step width factor α_t for iteration t. Given the current value of λ with optimal solutions $\tilde{x}(\lambda)$, $\tilde{y}^0(\lambda)$ and $\tilde{y}^1(\lambda)$, the gradient is computed as:

$$\nabla\tilde{g}(\lambda) = \tilde{y}^0(\lambda)' - \tilde{y}^1(\lambda)', \tag{6.19}$$

where $'$ denotes the transposed vector. Transposing the y vectors is necessary to match the dimensionality of λ in the update step.

With the split of the minimization problem, the efficient optimization strategies can be applied to the part problems: $\tilde{y}^0(\lambda)$ can be determined with the graph-cut algorithm and $\tilde{y}^1(\lambda)$ with PSM solution techniques. The solutions found with the PSM algorithms automatically satisfy the feasibility constraint.

The gradient ascent method can lead to solutions with decreased quality, so a list of the l best reached points, the dual solution candidates, is stored as $L = \{\lambda_1, \ldots, \lambda_l\}$. This leads to the following algorithm:

1. Obtain the body part hypotheses, calculate the edge weights and pixel foreground probabilities.
2. Set λ to an arbitrary starting value (not specified by Wang and Koller).
3. Use the gradient ascent to determine new λ values.

a) Optimize the two part problems given the current value of λ as follows and calculate the sum of the energies of the two part problems $E(\lambda)$:

$$\begin{aligned}\left(\tilde{x}(\lambda), \tilde{y}^0(\lambda)\right) &= \arg\min_{x,y^0} \left[E^0\left(x, y^0\right) + \lambda \cdot y^0\right], \leftarrow \text{Graph-cuts} \\ \tilde{y}^1(\lambda) &= \arg\min_{y^1 \in \mathcal{C}} \left[E^1\left(y^1\right) - \lambda \cdot y^1\right], \leftarrow \text{PSM solution alg.} \\ E(\lambda) &= E^0\left(\tilde{x}, \tilde{y}^0\right) + \lambda \cdot \tilde{y}^0 + E^1\left(\tilde{y}^1\right) - \lambda \cdot \tilde{y}^1. \end{aligned} \tag{6.20}$$

b) Compute the gradient as specified in Equation 6.19 and update λ as

$$\lambda \leftarrow \lambda + \alpha_t \cdot \nabla \tilde{g}(\lambda), \tag{6.21}$$

c) If $E(\lambda)$ is high enough, update the list L of dual solution candidates.

d) If a termination criterion is met, break. Otherwise, go to step (a).

4. Determine feasible solutions based on the dual solution candidates in L, calculate their primal energies and find the best solution. This process is explained in Section 6.3.3.

The termination criterion for step 3(d) is not specified by Wang and Koller; the step width α_t can be set adaptively per step as described in [KPT11]. The solutions created for the part minimization problems in step 3(a) take into account the respective energy terms and λ times the y-selection vector for the respective energy function (see Equation 6.18). Thus, the λ-value gives a weight resulting in a penalty or a bonus for selecting certain body part hypotheses or pixels.

Wang and Koller interpret this mechanism as message passing between the two part problems. The simplest case is that in an iteration the selection variables y_{jk} agree in both selections $\tilde{y}^0(\lambda)$ and $\tilde{y}^1(\lambda)$. In this case, the according component of the gradient $(\nabla \tilde{g}(\lambda))_{jk} = 0$ as specified in Equation 6.19 and the respective component of λ is not updated for the next iteration.

However, assuming that e.g. the part is selected by the pixel level optimization but not by the HPE optimization, i.e. $(\tilde{y}^0(\lambda))_{jk} = 1$, but $(\tilde{y}^1(\lambda))_{jk} = 0$, a "message" is passed. The value of the gradient is calculated as $(\nabla \tilde{g}(\lambda))_{jk} = 1$ and $(\lambda)_{jk}$ is updated accordingly. This results in an additional "penalty" for selecting the body part for the pixel level optimization and in an "encouragement" to select the body part for the HPE optimization (see Equation 6.18). The gradient ascent tries to pull the two copies of each part selection variable towards each other (compare to [WK11, p. 5]).

6.3.3 Construction of Primal Solutions

With the list of dual solution candidates, the one with the best overall energy value must be selected. To determine it, the residual energy term (as given in

Equation 6.18) is included in the calculations, so the optimal values $\bar{x}(\lambda)$ and $\bar{y}(\lambda)$ for $\bar{x}$ and $\bar{y}$ variables for the respective λ value must be determined.

Note that the $\tilde{y}^0(\lambda)$ and $\tilde{y}^1(\lambda)$ values do not have to agree in the proposed solutions from the dual optimization step. A preprocessing step is necessary to find solutions for $\bar{x}(\lambda)$ and $\bar{y}(\lambda)$ taking into account the hard constraint $\tilde{y}^0(\lambda) = \tilde{y}^1(\lambda)$.

Based on $\tilde{y}^1(\lambda)$: $\tilde{y}^1(\lambda)$ is the result of solving a PSM and automatically compliant with the feasibility constraint, so $\tilde{y}^1(\lambda) \in \mathcal{C}$, and $\bar{y}(\lambda)$ is defined as $\bar{y}(\lambda) = \tilde{y}^1(\lambda)$. Ignoring $\tilde{y}^0(\lambda)$, only the pixel selection $\tilde{x}(\lambda)$ must be updated with respect to $\bar{y}(\lambda)$ to determine $\bar{x}(\lambda)$.
This can be done efficiently with another run of graph-cut with $\bar{y}(\lambda)$ as as body part selection. This time, graph-cut can be run on the primal energy function as specified in Equation 6.2, since E^{r2} and $E^{\text{part-pixel}}$ only depend on the pixel selection and the other terms are constant when y is fixed. This solution can be better than the former $\tilde{x}(\lambda)$ proposed by the part problem, since it uses the information of the residual energy term.

Based on $\tilde{y}^0(\lambda)$: The selection specified by $\tilde{y}^0(\lambda)$ does not necessarily comply with the feasibility constraint, so it could be that $\tilde{y}^0(\lambda) \notin \mathcal{C}$. However, when the pixel selection is fixed, $E^{\text{part-pixel}}$ only depends on the body part selection. Thus, the solution steps for the PSM can be evaluated again, this time including the energy terms $E^{\text{part-pixel}}$, $E^{\text{part-single}}$ and E^{joints} to find the best body part configuration. With the determined $\bar{y}(\lambda)$ from this process, the residual energy term is included in the final solution by doing the same steps as in the explanation above to find an optimized $\bar{x}(\lambda)$.

The results of both methods are stored for each $\lambda_i \in L$ values and finally, the one with the lowest primal energy is selected. Note that the additional energy terms in E^{residual} are not used to select the dual solution candidates. They are only evaluated on the few, proposed values resulting from that step. Thus, it is possible to integrate intractable energy terms into the solution process which can not be minimized efficiently. It is sufficient that they can be evaluated efficiently. However, "[...] the quality of the primal solutions constructed can be affected by the fact that [Wang and Koller] are maximizing the relaxed dual function" [WK11, p. 5]. The algorithm performance is discussed in the final section of this chapter.

6.4 Results

6.4.1 Evaluation

Wang and Koller use the pose estimation approach of Andriluka et al. [ARS09] to find the top ten hypotheses for each body part. Depending on the hypothesis providing method, very different results can be expected. The comparison to the approach by Andriluka et al. is especially interesting, since it shows the achieved

improvements. For the evaluation, a body part is considered as correct, if it has less than half of its length as distance to the ground truth location. The performance is evaluated on the dataset provided by Ramanan [Ram07].

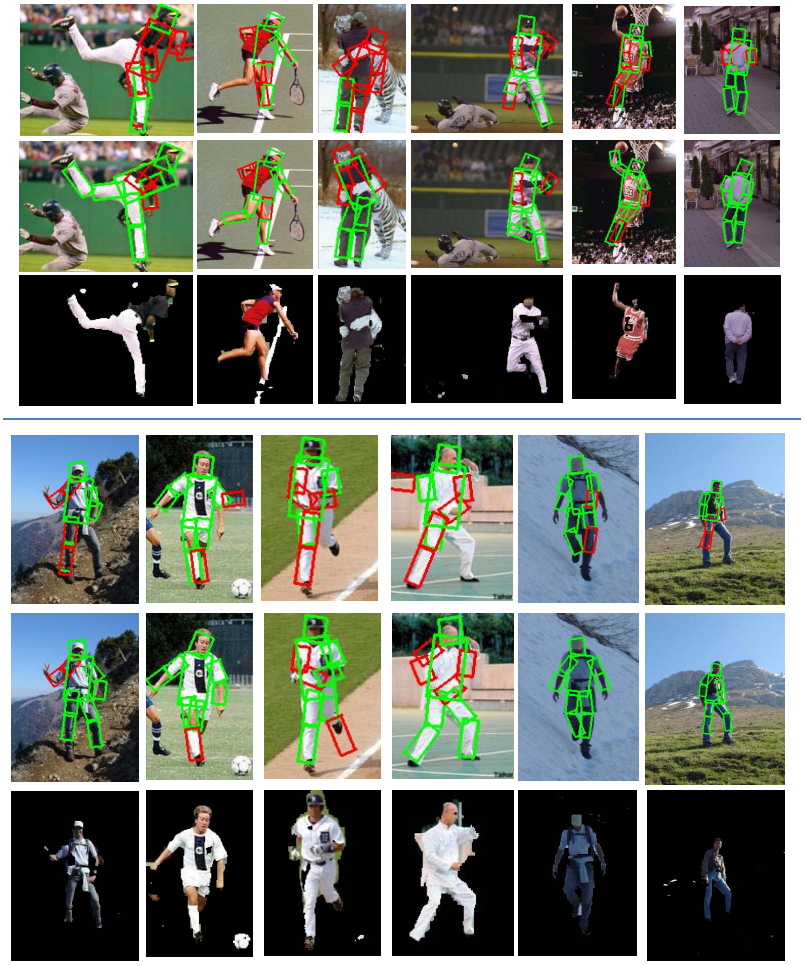


Figure 6.5: Qualitative comparison of the results of Andriluka et al. [ARS09] and Wang and Koller's method [WK11] (see [WK11, p. 7]). For each image, **top:** results of [ARS09], **middle:** pose estimation with [WK11], **bottom:** segmentation results with [WK11].

On this dataset, the approach of Wang and Koller achieves an accuracy of 61.51%. This is an increase of 6.31% compared to the performance of the approach of Andriluka et al. [ARS09] that provides the hypotheses. The top ten hypotheses for each body part allow a theoretical localization accuracy of 77.02% on this test set for a perfect selection of parts (see [WK11, p. 6]). A detailed comparison of the achieved localization rates by both approaches, along with a comparison to the approach by Johnson and Everingham presented in Chapter 5, is given in Table 5.3b. Especially the improvement of the leg localization rate compared to the method of Andriluka et al. is striking.

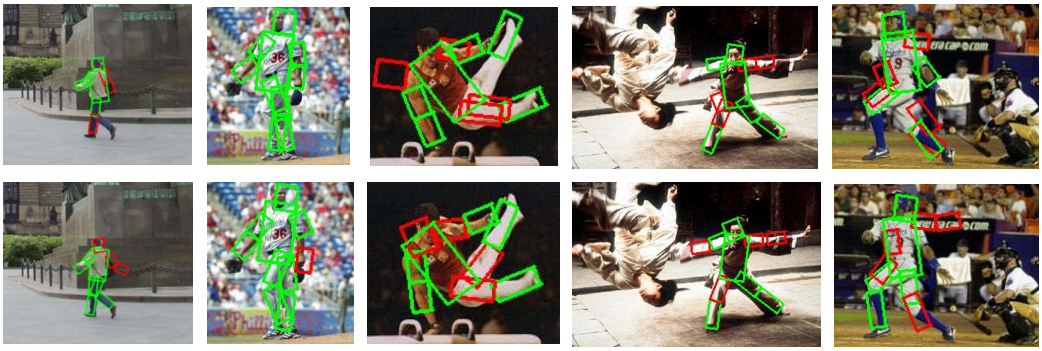


Figure 6.6: Illustration of the limitations of the evaluation metric [WK11, p. 8]. **Top row:** results of [ARS09], **bottom row:** results of [WK11]. The improved localization of boxes is not reflected in the qualitative analysis.

Figure 6.5 shows example images with the results of the method of Andriluka et al. [ARS09] and the results of Wang and Koller [WK11] on the same images. The localization boxes are color coded green and red to reflect correctly and incorrectly detected body parts according to the evaluation criterion.

Some of the improvements achieved by Wang and Kollers method are not reflected in the quantitative analysis, since a better positioning of already "true" labeled hypotheses has no effect. This fact is illustrated in Figure 6.6.

6.4.2 Discussion

Wang and Koller show that it is possible to combine knowledge from different levels of information efficiently. Even intractable energy terms that can occur in models spanning multiple levels can be handled by their framework. However, they do not give an indication on runtime performance. The multiple usage of graph-cut and PSM solution algorithms has considerable computational cost, rendering the approach unusable for real-time scenarios. Another limitation is the restriction to pose estimation for one person. Hypotheses for multiple persons can not be handled yet.

Wang and Koller suggest in their closing section to extend the approach to handle video sequences by adding additional potentials and extend the energy terms. This extension seems straightforward for the method and they expect to achieve good results for tracking. They also show the limitations of the current standard evaluation criterion for correct and incorrect part localizations. A new evaluation criterion could help to make better annotations visible in a qualitative analysis.

7 Conclusion

Four different approaches to HPE have been analyzed in the four preceding chapters. They were chosen such, that during their analysis many of the frequently applied techniques for HPE could be explained. The following section contains ideas on how some of these "parts" can be reassembled to form a new approach to HPE, followed by a general outlook.

7.1 Reassembling the Parts

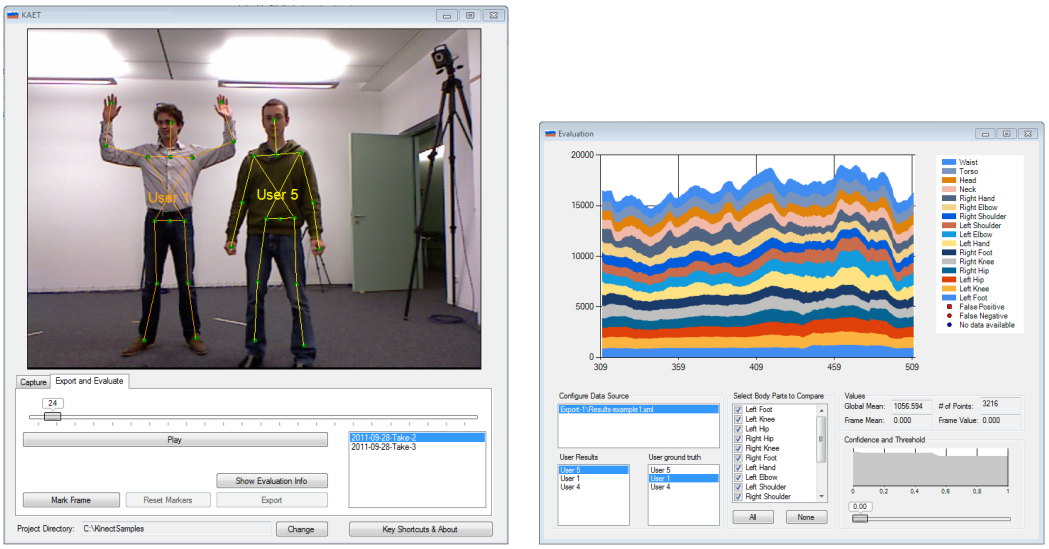
7.1.1 Dataset Acquisition

The necessity for a large training dataset has been discussed during the task analysis in Section 2.1 and during the analyses of the approaches. The variety of human shape and pose, skin color and clothing color and the problem of missing occlusion and depth data require comprehensive training sets. Johnson and Everingham found with the use of AMT an elegant way to obtain annotations for many hand-selected images (see Chapter 5). With their approach even low quality annotations can be made useful for classifier training.

Since the launch of the Kinect a reliable and cheap HPE system is available. It provides foreground/background segmentation data additional to the estimated joint positions and a CMOS camera stream. Consequently, it can be used to obtain a large dataset with reliable pose annotations and segmentation information. With this data, it can accelerate the process of developing a HPE system working only with 2D information.

This idea has led to the development of the "Kinect Annotation and Evaluation Tool" (KAET) [Las11]. It streamlines the task of taking video sequences with depth, segmentation, image and pose information, extracting the appropriate frames and export them for classifier training steps (see Figure 7.1). In a second step, it also allows for evaluation of a HPE approach, by visualizing a comparison of the recorded annotations of the Kinect and the ones provided by the new approach.

Using this program, a large set of images with intermediate quality annotations can be collected. The iterative annotation improvement approach by Johnson and Everingham (described in Section 5.3) might still be useful to correct for smaller



(a) Take-management dialog. **(b)** Evaluation tools included in KAET.

Figure 7.1: KAET tool user interface.

inaccuracies. To maximize the benefit of the pose data, a 'furthest neighbor' clustering could be applied to images from each sequence, similar to the approach by Shotton et al. (see Section 3.3.3).

However, the use of the Kinect is restricted to indoor environments. A possibility to avoid this limitation is, to substitute the background of images automatically with other environments. It remains to be seen, whether this approach can incorporate a sufficient amount of variance to the data. If not, the approach by Johnson and Everingham remains a possibility to get many additional annotations. In the latter scenario, the data from the Kinect can be used as the needed "expert" ground truth, maybe in combination with motion capture data.

7.1.2 Pose Estimation

The most reliable pose estimation approach known to this date is the Kinect system by Shotton et al. [SFC+11] (described in Chapter 3), which uses 3D scene information. An interesting option is, to apply similar techniques to 2D color images. In their work Shotton et al. mention, that their approach reaches 0.465 mean average precision on silhouette images (a binary image consisting of labels for foreground and background, but no further information) [SFC+11, p. 6].

When applied to standard 2D color images, the silhouette or segmentation of the person is not available. On the other hand, the color information could be used to improve the classifier accuracy. The feature set f_θ could be replaced with a feature set working on colors. It could be chosen e.g. as difference measure in the HSL

colorspace, to mainly capture differences in hue and lightness. It must be verified experimentally whether this setup can reach satisfying performance. Even if not, it could be used as a per pixel information provider for a combined approach, similar to the one by Wang and Koller described in Chapter 6.

Taking Wang and Kollers approach as starting point, it could be enhanced with clustered models as proposed by Johnson and Everingham (see Chapter 5). Either the PSM could be completely replaced by Clustered PSMs, or only the mixture of linear SVMs could be used to provide better body part hypotheses.

Another enhancement could be added by including a new energy function for object detector information. This way, the context could be integrated into the reasoning process, similar to the approach by Singh et al. analyzed in Chapter 4.

After all, this theoretical work can give ideas for the design of new methods, but their performance can hardly be predicted. Some of the aforementioned suggestions will be implemented for further evaluation in future work.

7.1.3 Evaluation

The currently most frequently used evaluation criterion is based on the "box" representation of body parts (see e.g. Figure 6.6). A detection is considered correct, if the detected box overlaps with the ground truth box for more than 50%. This avoids problems with inaccurate annotations, since it allows for slight shifts of the box, still counting a detection as true. However, at the same time it ignores slight improvements of detections. Wang and Koller state, that "[...] in many cases our improvement is not reflected quantitatively due to the limitation of the evaluation criterion" [WK11, p. 6].

This problem could be addressed, by introducing an evaluation strategy that does not only count "true" and "false" annotations, but includes a measure for the distance to the ground truth annotation. For this purpose, we propose to use a joint annotation (as visible in Figure 7.1a) instead of box annotations.

For each joint detection, the distance to the ground truth annotation is counted. From this data, a cumulative chart similar to the one in Figure 7.2 is created.

The chart shows, how many detections are lying within the given pixel range from the ground truth. The maximum distance in the chart should be fixed, e.g. to 15 pixels, to allow for direct comparison of several evaluation charts. The distance between ground truth and annotation location should be measured with city-block distance to avoid rounding issues, i.e.

$$d\left(\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}\right) = \|x_2 - x_1\| + \|y_2 - y_1\| \tag{7.1}$$

An approach is considered to have better performance, if the resulting curve is closer to the top-left corner of the chart and if the space above the curve is smaller

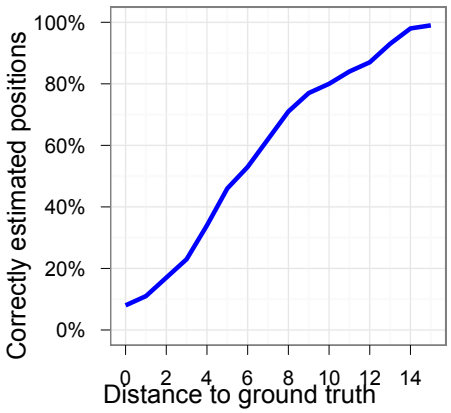


Figure 7.2: Example for the suggested evaluation plot with artificial data.

(similar to Receiver Operating Characteristic curves). This could provide a fair and tough comparison for methods evaluated on the same dataset. If necessary, a similar evaluation method could be created for box annotations, by counting correct annotations per overlapping level (e.g. x-axis: 100%, 95%, 90%, ... overlapping, y-axis: number of boxes considered as correct).

7.2 Outlook

The work group of the BlueBrain project [Blu12] works since six years on simulating a complete human brain on molecular level. The group uses an IBM BlueGene supercomputer for simulation - and could already present first results. It currently applies for a "Future and Emerging Technologies" research grant of 1 billion Euro of the European Union to reach their aim until the year 2023 [Nat11, Mar11]. By simulating the brain, the researchers hope to find answers to questions regarding the source of consciousness and understanding, as well as learning.

With their results, also machine learning could change. But this will take additional time and it is unclear, whether their findings lead to algorithms realizable on "small" computers for everyday use. Contrary, the fast progress in the field of HPE kindles the hope for soon, satisfactory results. It remains an exciting race to the first big success in a general 2D color image setting.

Acknowledgments

Many thanks go to my supervisor Thomas Greif and my supervising reviewer Prof. Rainer Lienhart. They took a lot of time for tips, suggestions for improvements and fruitful discussions.

I am very grateful to my friends Jenny Hüttner, Richard Schreiber, Manuel Deil, Ferenc Bujtor and Stephan Kammerer for working thoroughly through the non-final version of this thesis and pointing out many possibilities for improvements.

Thanks to the many authors who make their articles freely accessible on the web. Due to them, many of the referenced documents could be linked in the bibliography of the digital edition for the readers convenience.

Bibliography

[ARS09] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial Structures Revisited: People Detection and Articulated Pose Estimation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5206754. (Cited on pages 17, 50, 68, 73, 77, 84, 85, 86 and 100.)

[Bis06] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. (Cited on pages 12, 14, 15, 16, 62, 74 and 99.)

[BJ01] Yuri Y. Boykov and Marie-Pierre Jolly. Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images. *Inernational Conference on Computer Vision*, I(July): 105–112, 2001. URL: http://www.csd.uwo.ca/~yuri/Papers/iccv01.pdf. (Cited on pages 71, 77 and 79.)

[Blu12] The Blue Brain Project, URL: http://bluebrain.epfl.ch/. (Cited on pages 10 and 90.)

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. (Cited on pages 80 and 100.)

[CM02] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5): 603–619, 2002. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1000236. (Cited on page 36.)

[CV95] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 297(20): 273–297, 1995. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.15.9362. (Cited on page 14.)

[CVP10] CVPR 2010: IEEE Conference on Computer Vision and Pattern Recognition Homepage - Paper Submission Results, URL: http://cvl.umiacs.umd.edu/conferences/cvpr2010/results/. (Cited on page 2.)

[CVP11] CVPR 2011: IEEE Conference on Computer Vision and Pattern Recognition Homepage - Statistics, URL: http://cvpr2011.org/statistics. (Cited on page 2.)

[DT05] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. *IEEE Conference on Computer Vision and Pattern*

Recognition, 1: 886–893, 2005. URL: http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf. (Cited on pages 10, 11, 51 and 99.)

[EG09] M. Enzweiler and D.M. Gavrila. Monocular Pedestrian Detection: Survey and Experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12): 2179 –2195, 2009. URL: http://dx.doi.org/10.1109/TPAMI.2008.260. (Cited on page 4.)

[FE73] Martin A. Fischler and Robert A. Elschlager. The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, 100(1): 67–92, January 1973. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1672195. (Cited on pages 17, 18 and 99.)

[FGMR10] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9): 1627–45, September 2010. URL: http://www.ncbi.nlm.nih.gov/pubmed/20634557. (Cited on page 17.)

[FH00] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Matching of Pictorial Structures. *IEEE Conference on Computer Vision and Pattern Recognition*, 2: 66–73, 2000. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=854739. (Cited on pages 17 and 22.)

[FH05] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial Structures for Object Recognition. *International Journal of Computer Vision*, 61(1): 55–79, January 2005. URL: http://www.springerlink.com/index/m441rlt8j0063k7k.pdf. (Cited on pages 17, 21, 23, 24, 43, 50, 53, 59 and 99.)

[FHT00] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive Logistic Regression: A Statistical View of Boosting. *The Annals of Statistics*, 28(2): 337–407, 2000. URL: http://www.jstor.org/stable/2674028. (Cited on page 50.)

[FP02] David Forsyth and John Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2002. (Cited on pages 6, 7 and 99.)

[FRK08] Zhouyu Fu and Antonio Robles-Kelly. On Mixtures of Linear SVMs for Nonlinear Classification. *Lecture Notes in Computer Science*, 5342/2008: 489–499, 2008. URL: http://dx.doi.org/10.1007/978-3-540-89689-0_53. (Cited on page 62.)

[FS97] Yoav Freund and Robert E. Schapire. A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1): 119–139, 1997. URL: http://www.cs.princeton.edu/~schapire/uncompress-papers.cgi/FreundSc95.ps. (Cited on page 17.)

[fut11] futurepicture.org. Images of the Kinect Speckle Pattern, URL: http://www.futurepicture.org/?p=129. (Cited on pages 31 and 99.)

[GKD09] Abhinav Gupta, Aniruddha Kembhavi, and Larry S. Davis. Observing Human-Object Interactions: Using Spatial and Functional Compatibility for Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10): 1775–89, October 2009. URL: http://www.ncbi.nlm.nih.gov/pubmed/19696449. (Cited on pages 51 and 53.)

[GPKT10] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real Time Motion Capture Using a Single Time-Of-Flight Camera. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 755–762, 2010. URL: http://ai.stanford.edu/~koller/Papers/Ganapathi+al:CVPR10.pdf. (Cited on pages 38 and 99.)

[Gra99] Anthony C. Grayling. *Philosophy 1: A Guide through the Subject*. Oxford University Press, 1999. (Cited on page 9.)

[Gui11] Guinness World Records. Kinect Confirmed As Fastest-Selling Consumer Electronics Device, URL: http://community.guinnessworldrecords.com/_Kinect-Announced-As-Fastest-Selling-Consumer-Electronics-Device/blog/3376939/7691.html. (Cited on page 1.)

[iFi12] iFixIt Kinect Teardown, URL: http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/1. (Cited on page 29.)

[JÖ5] Bernd Jähne. *Digital Image Processing*. Springer, Heidelberg, 6th editio edition, 2005. (Cited on page 8.)

[JE10] Sam Johnson and Mark Everingham. Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation. In *Proceedings of the 21st British Machine Vision Conference*, number I, 2010. (Cited on pages 25, 57, 58, 65, 67, 68 and 69.)

[JE11] Sam Johnson and Mark Everingham. Learning Effective Human Pose Estimation from Inaccurate Annotation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. URL: http://www.comp.leeds.ac.uk/me/Publications/cvpr11_pose.pdf. (Cited on pages 1, 55, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70 and 100.)

[Kin10] Kinect Homepage, URL: http://www.xbox.com/kinect. (Cited on pages 1 and 27.)

[Kin12] Kinect for Windows Team. Near Mode: What it is (and isn't), URL: http://blogs.msdn.com/b/kinectforwindows/archive/2012/01/20/near-mode-what-it-is-and-isn-t.aspx. (Cited on pages 28 and 99.)

[Koh95] Ron Kohavi. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference on*

Artificial Intelligence, 1995. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.133.9187. (Cited on pages 16 and 17.)

[KPT11] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF Energy Minimization and Beyond via Dual Decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3): 531–52, March 2011. URL: http://www.ncbi.nlm.nih.gov/pubmed/20479493. (Cited on page 83.)

[KZ04] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2): 147–159, February 2004. URL: http://www.ncbi.nlm.nih.gov/pubmed/15376891. (Cited on pages 71 and 79.)

[Las11] Christoph Lassner. Kinect Annotation and Evaluation Tool, URL: http://www.multimedia-computing.org/wiki/KAET. (Cited on page 87.)

[Mar03] Horst de Marées. *Sportphysiologie*. Sportverlag Strauss, 2003. (Cited on page 9.)

[Mar11] Henry Markram. International Supercomputing Conference 2011: Keynote "Simulating the Brain - The Next Decisive Years", URL: http://lecture2go.uni-hamburg.de/konferenzen/-/k/12293. (Cited on page 90.)

[Mer12] Merriam-Webster Medical Dictionary: Neuroscience, URL: http://www.merriam-webster.com/medlineplus/neuroscience. (Cited on page 9.)

[MG01] Thomas B. Moeslund and Erik Granum. A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding*, 81(3): 231–268, March 2001. URL: http://linkinghub.elsevier.com/retrieve/pii/S107731420090897X. (Cited on page 3.)

[MHK06] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3): 90–126, November 2006. URL: http://linkinghub.elsevier.com/retrieve/pii/S1077314206001263. (Cited on pages 2 and 3.)

[Mit97] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997. (Cited on pages 12, 13 and 14.)

[Nat11] Nature. European researchers chase billion-euro technology prize, URL: http://www.nature.com/news/2011/110308/full/news.2011.143.html. (Cited on page 90.)

[Oxf12] Oxford Dictionaries: Physics, URL: http://oxforddictionaries.com/definition/physics?q=physics. (Cited on page 9.)

[Pop07] Ronald Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1-2): 4–18, October 2007. URL: http://linkinghub.elsevier.com/retrieve/pii/S1077314206002293. (Cited on page 3.)

[Pop10] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6): 976–990, June 2010. URL: http://linkinghub.elsevier.com/retrieve/pii/S0262885609002704. (Cited on page 4.)

[Pri12a] PrimeSense - 3D sensing technology, URL: http://www.primesense.com/en/technology/115-the-primesense-3d-sensing-solution. (Cited on pages 30 and 99.)

[Pri12b] PrimeSense - Processor Technology, URL: http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/2. (Cited on pages 29 and 99.)

[Ram07] Deva Ramanan. Learning to parse images of articulated bodies. *Advances in Neural Information Processing Systems*, pages 1129–1136, 2007. URL: http://www.ics.uci.edu/~dramanan/papers/parse.pdf. (Cited on pages 25, 43, 49, 50, 51, 52, 53, 58, 68, 69 and 85.)

[RS06] Deva Ramanan and Cristian Sminchisescu. Training Deformable Models for Localization. *IEEE Conference on Computer Vision and Pattern Recognition*, 1: 206–213, 2006. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1640761. (Cited on page 50.)

[SF08] Alexander Sorokin and David Forsyth. Utility data annotation with Amazon Mechanical Turk. *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, (c): 1–8, 2008. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4562953. (Cited on page 56.)

[SFC+11] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-Time Human Pose Recognition in Parts from Single Depth Images. *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. URL: http://www.cs.dartmouth.edu/~cs104/BodyPartRecognition.pdf. (Cited on pages 25, 27, 28, 31, 32, 34, 35, 37, 38, 39, 40, 55, 88 and 99.)

[SFCB11] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, and Andrew Blake. CVPR 2011 Talk - Real-time Human Pose Recognition in Parts from Single Depth Images, URL: http://techtalks.tv/talks/54235/. (Cited on pages 28, 37 and 99.)

[SKN10] Vivek Kumar Singh, Furqan M.. Khan, and Ram Nevatia. Multiple Pose Context Trees for estimating Human Pose in Object Context. *IEEE*

Conference on Computer Vision and Pattern Recognition Workshops, pages 17–24, 2010. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5543186. (Cited on pages 25, 41, 42, 44, 48, 49, 50, 51, 52, 53, 99 and 100.)

[Sta12] Manfred Stader. 3-D street painting, URL: http://www.3d-street-art.com/3dstreetpainting_asianpaints.htm. (Cited on pages 8 and 99.)

[Sze10] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010. (Cited on page 8.)

[Via11] Mikkel Viager. Analysis of Kinect for mobile robots, URL: http://www.scribd.com/doc/56470872/Analysis-of-Kinect-for-Mobile-Robots-unofficial-Kinect-data-sheet-on-page-27. (Cited on pages 30 and 31.)

[Wik12] Wikipedia. Lotus effect, URL: http://en.wikipedia.org/wiki/Lotus_effect. (Cited on page 10.)

[WK11] Huayan Wang and Daphne Koller. Multi-level inference by relaxed dual decomposition for human pose segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2433–2440, 2011. URL: http://www.robotics.stanford.edu/~koller/Papers/Wang+Koller:CVPR11.pdf. (Cited on pages 25, 68, 71, 72, 73, 75, 79, 81, 83, 84, 85, 86, 89 and 100.)

[WRB11] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2): 224—-241, 2011. URL: http://www.sciencedirect.com/science/article/pii/S1077314210002171. (Cited on page 4.)

[WY09] Wei Wei and An Yunxiao. Vision-Based Human Motion Recognition: A Survey. *Second International Conference on Intelligent Networks and Intelligent Systems*, pages 386–389, November 2009. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5366975. (Cited on page 4.)

[Zec11] Dan Zecha. Detecting Swimmers and Estimating their Pose in a Swimming Channel. 2011. URL: http://www.multimedia-computing.de/mediawiki//images/4/4e/BA_DanZecha.pdf. (Cited on page 3.)

[ZH08] Huiyu Zhou and Huosheng Hu. Human motion tracking for rehabilitation-A survey. *Biomedical Signal Processing and Control*, 3(1): 1–18, January 2008. URL: http://linkinghub.elsevier.com/retrieve/pii/S1746809407000778. (Cited on pages 3 and 4.)

List of Figures

Nomenclature

AMT	Amazon Mechanical Turk, https://www.mturk.com/mturk/welcome.
CCD	"A charge-coupled device (CCD) is a device for the movement of electrical charge, usually from within the device to an area where the charge can be manipulated, for example conversion into a digital value. This is achieved by shifting the signals between stages within the device one at a time. CCDs move charge between capacitive bins in the device, with the shift allowing for the transfer of charge between bins. The CCD is a major technology for digital imaging." [http://en.wikipedia.org/wiki/Charge-coupled_device]
CMOS	"Complementary metal–oxide–semiconductor (CMOS) is a technology for constructing integrated circuits. CMOS technology is used in microprocessors, microcontrollers, static RAM, and other digital logic circuits. CMOS technology is also used for several analog circuits such as image sensors (CMOS sensor), data converters, and highly integrated transceivers for many types of communication. Frank Wanlass patented CMOS in 1967 (US patent 3,356,858)." [http://en.wikipedia.org/wiki/CMOS]
CMOS sensor	"An active-pixel sensor (APS) is an image sensor consisting of an integrated circuit containing an array of pixel sensors, each pixel containing a photodetector and an active amplifier. There are many types of active pixel sensors including the CMOS APS used most commonly in cell phone cameras, web cameras and in some DSLRs. Such an image sensor is produced by a CMOS process (and is hence also known as a CMOS sensor), and has emerged as an alternative to charge-coupled device (CCD) image sensors." [http://en.wikipedia.org/wiki/Active_pixel_sensor]
HIT	Amazon Mechanical Turk Task.
HOG	Histogram of Oriented Gradients. A point descriptor explained in Section 2.3.1.
PSM	Pictorial Structure Model. A model for deformable objects explained in Section 2.3.4.

HPE	Human Pose Estimation.
HSL	Hue, Saturation and Lightness. A cylindrical colorspace can be defined on these three values.
PCT	Pose Context Tree. A tree-structured graphical model explained in Chapter 4.
SVM	Support Vector Machine. A binary classifier frequently used in combination with HOG descriptors. Explained in Section 2.3.2.2.

Appendix

A Digital Edition

B Eidesstattliche Erklärung

Ich versichere, dass die Diplomarbeit von mir selbständig verfasst wurde und dass ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Zitate habe ich klar gekennzeichnet.

Augsburg, den 7. Mai 2012

Christoph Lassner