Fast Tracking of Hand and Finger Articulations Using a Single Depth Camera

Srinath Sridhar, Antti Oulasvirta,
Christian Theobalt

MPI–I–2014–4–002 October 2014

Authors' Addresses

Srinath Sridhar
Max Planck Institute for Informatics
Campus E 1 4

D-66123 Saarbrücken, Germany

Antti Oulasvirta
Electrical Engineering Building
Department of Communications and Networking

Otakaari 5, 13000 Aalto University, Finland

Christian Theobalt
Max Planck Institute for Informatics
Campus E 1 4

D-66123 Saarbrücken, Germany

Acknowledgements

This work was supported by the ERC Starting Grant CapReal. We would like to thank Franziska Müller and Christian Richardt.

Abstract

Using hand gestures as input in human–computer interaction is of ever-increasing interest. Markerless tracking of hands and fingers is a promising enabler, but adoption has been hampered because of tracking problems, complex and dense capture setups, high computing requirements, equipment costs, and poor latency. In this paper, we present a method that addresses these issues. Our method tracks rapid and complex articulations of the hand using a single depth camera. It is fast (50 fps without GPU support) and supports varying close-range camera-to-scene arrangements, such as in desktop or egocentric settings, where the camera can even move. We frame pose estimation as an optimization problem in depth using a new objective function based on a collection of Gaussian functions, focusing particularly on robust tracking of finger articulations. We demonstrate the benefits of the method in several interaction applications ranging from manipulating objects in a 3D *blocks world* to egocentric interaction on the go. We also present extensive evaluation of our method on publicly available datasets which shows that our method achieves competitive accuracy.

Keywords

Hand tracking, human pose estimation, human–computer interaction, input strategy

Fast Tracking of Hand and Finger Articulations Using a Single Depth Camera

Srinath Sridhar[*]
Max Planck Institute for Informatics

Antti Oulasvirta[†]
Aalto University

Christian Theobalt[‡]
Max Planck Institute for Informatics

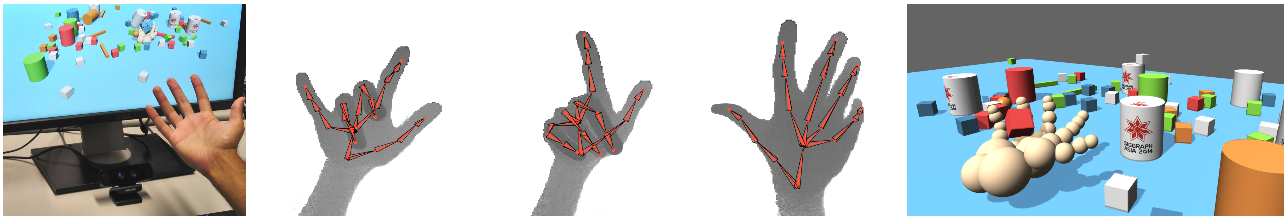

Figure 1: *We present a novel method for realtime hand tracking using a single depth camera. We show that our method is suitable for interaction applications involving fast and subtle finger articulations. It allows use in interactive applications with different camera views, such as virtual object manipulation on a desktop.*

Abstract

Using hand gestures as input in human–computer interaction is of ever-increasing interest. Markerless tracking of hands and fingers is a promising enabler, but adoption has been hampered because of tracking problems, complex and dense capture setups, high computing requirements, equipment costs, and poor latency. In this paper, we present a method that addresses these issues. Our method tracks rapid and complex articulations of the hand using a single depth camera. It is fast (50 fps without GPU support) and supports varying close-range camera-to-scene arrangements, such as in desktop or egocentric settings, where the camera can even move. We frame pose estimation as an optimization problem in depth using a new objective function based on a collection of Gaussian functions, focusing particularly on robust tracking of finger articulations. We demonstrate the benefits of the method in several interaction applications ranging from manipulating objects in a 3D *blocks world* to egocentric interaction on the go. We also present extensive evaluation of of our method on publicly available datasets which shows that our method achieves competitive accuracy.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Input Devices and Strategies

Keywords: hand tracking, 3D interaction, realtime tracking

1 Introduction

Exploiting the exceptional dexterity of the human hand for computer input is a prime goal for research in human–computer interaction (HCI). The human hand has 26 degrees of freedom (DOF), only a few of which are exploited by conventional input devices such as the mouse. Even the widely popular multi-touch displays capture only the 2D positions and gestures of fingertips. Similarly in computer graphics, interactive techniques are gaining more importance. Reliable and easy-to-use hand tracking would enable new ways of expressing creativity in model or animation design, or real-time character control. Contact- based and marker-based methods have been used in the past to capture hand articulation for virtual reality applications [Zimmerman et al. 1986; Sturman and Zeltzer 1994]. However, methods that can be used outside of motion capture studios while still being robust and fast remain elusive.

Markerless, non-contact methods for hand tracking are preferable because they do not constrain the free motion of fingers. But realtime vision-based tracking of hands presents several unique challenges. First, natural hand gestures involve control of several DOFs simultaneously, fast motions, rapid changes in direction, and self-occlusions. Tracking fast *finger articulations* at *high framerates* and *low latency* is critical for many interaction scenarios, but has remained a challenge even for state-of-the-art trackers. Important articulations include adduction/abduction, apposition/opposition (pinch), caging/fisting the palm, and flexing/extending fingers. Second, *setup costs* including the number of cameras used should be low, as this directly affects adoption for consumer applications. Finally, the ubiquity of interaction requires tracking in many different setups including desktop, laptop, mobile and wearable configurations.

This paper presents a novel method for hand tracking that aims to address these challenges. Our model-based method is capable of tracking the hand robustly at high framerates (50 fps without GPU), and deals efficiently with even complex and fast finger motions with notable occlusions. The method is easy to set up and use, because we only necessitate a single depth camera that is allowed to move relative to the scene during capture. Unlike many multi-camera methods that require calibration, our method supports varying close-range camera-to-hand arrangements including desktop, egocentric or mobile settings. The high framerate achieved by our method ensures tracking of fast motions and low latency for interaction applications.

The results and performance are made possible by a novel, efficient representation of both input *depth* and the hand as a collection of Gaussian functions. This representation allows us to formulate pose estimation as a new 2.5D optimization problem in depth. To this end, we define an objective function that maximizes the similarity of the input depth with a *kinematic* hand model, and uses additional prior and data terms that avoid finger collisions and that preserve the smoothness of reconstructed motions. Because we use the Gaussian representation, our objective function and its derivative have an analytic formulation that enables rapid optimization. We are thus able to achieve fast convergence and high accuracy necessary for common interaction applications. Moreover, because we use a model-based approach, we can track complex articulations with occlusions. The generality of our method allows tracking of two hands and additional objects if necessary.

In order to demonstrate that our method is suitable for interac-

[*] ssridhar@mpi-inf.mpg.de
[†] antti.oulasvirta@aalto.fi
[‡] theobalt@mpi-inf.mpg.de

tion applications, we show two challenging applications that use hand tracking as input: (1) desktop interactions in a *blocks world* for virtual content manipulation, and (2) egocentric interaction for mobile settings. The scenarios show the robustness of our method under very different acquisition setups, and are challenging since complex and fast, but also fine-grained finger articulations need to be captured accurately from different camera perspectives and under different arm orientations.

1.1 Contributions

- A novel method for fast tracking of complex hand and finger articulations with notable occlusions using a single depth camera from varying close-range viewpoints.
- A new analytically differentiable objective function for pose optimization that allows accurate and realtime pose estimation from depth data. It relies on a new formulation of the depth and 3D hand geometry using a collection of Gaussian functions.
- An automatic method to create a personalized hand model for a user in under a second.
- Demonstrators of fast hand articulation input in challenging usage scenarios with different camera viewpoints.

In addition to qualitative experiments, we also performed extensive evaluation of our method on publicly available datasets and compare our results with other tracking methods.

2 Related Work

Free-hand tracking for interaction has been studied for many years with active work dating back to 1980 [Bolt 1980; Athitsos and Sclaroff 2003]. [Erol et al. 2007] provide an overview of methods until 2007. The introduction of consumer depth sensors has resulted in advancements in realtime hand tracking. Hand tracking is also closely related to full-body tracking and many parallels can be found in their algorithmic recipes [Baak et al. 2011; Shotton et al. 2011; Ganapathi et al. 2012; Kurmankhojayev et al. 2013]. However, we restrict our discussion to hand tracking. We categorize related work based on the most defining aspect of a particular method, although some methods may have overlapping features.

Gloves and Markers Marker-based systems rely on retro-reflective markers embedded on gloves to track the hand [Zimmerman et al. 1986; Sturman and Zeltzer 1994]. The 3D position of these markers is estimated using a multi-camera setup from which a full kinematic skeleton pose is reconstructed using inverse kinematics. Such methods are fast but require expensive equipment and restrict the free motion of fingers. To overcome the costs [Wang and Popovic 2009] proposed a color-glove for realtime tracking from a single RGB camera. They created a database of hand poses and find the nearest neighbor that best matches the input images. However, this method still requires users to wear a glove.

Multiple Views Multiple cameras provide a means to overcome pose estimation errors due to finger self-occlusions. [Oikonomidis et al. 2011a] proposed a method for tracking hands and objects together using a multi-camera setup. [Ballan et al. 2012] proposed a method for tracking hands in a constrained environment with good accuracy. [Wang et al. 2013] presented a method for capturing hand manipulations through motion control. However, all these methods are slow and are not suited for interactive applications. [Wang et al. 2011] proposed a multi-camera setup and a method to track hands without gloves at realtime speeds. Recently, [Sridhar et al. 2013] introduced a multi-view method that could track the hand at 10 fps. However, multi-camera systems are hard to set up and calibrate which may prevent them from being adopted for interaction.

Single Depth Camera The introduction of commercial depth sensors has resulted in a swathe of methods that make use of the depth information effectively. [Oikonomidis et al. 2011b] proposed a model-based method for tracking hand that made use of particle-swarm optimization. This method required GPU acceleration to achieve 15 fps and also depends on skin color segmentation which is sensitive to lighting. [Melax et al. 2013] proposed a method for tracking hands directly in depth by efficient parallel physics simulations.

Randomized decision forests have been used with great success for full body tracking [Shotton et al. 2011]. Many hand tracking methods adopt a similar strategy for pose estimation with varying success. [Keskin et al. 2011] proposed a method for recognizing finger spelling in depth data by training a decision forest. [Tang et al. 2013; Xu and Cheng 2013] also proposed methods based on regression forests. However, these methods require large amounts of training data and it is unclear how well they generalize to different users.

Applications There has been considerably less work in using tracked hand motion for interaction applications. [Wang et al. 2011] demonstrated a 3D CAD assembly task that used hands for input. But the interactions were restricted to 6 DOFs where finger articulations consisted mostly of pinching. Hand tracking was used by [Zhao et al. 2013] in a motion control system for grasping virtual objects to compensate for the lack of haptic feedback.

In this paper, we present a novel method for realtime (50 fps) hand and finger tracking that uses only a single depth camera. We also demonstrate the suitability of our method for interaction on many applications which use complex finger articulations for interactions.

3 Overview

The goal of our method is to robustly track the motion of hand and finger articulations given input from a single depth camera. We assume that no extrinsic calibration information about the camera is available. We also aim to achieve high framerates to capture, in particular, very fast finger articulations while ensuring low latency for applications, as well as high accuracy. Tracking hands is a hard problem because of the many self-occlusions, quick changes in speed and direction, uniform color distribution, and the large number of degrees of freedom.

In order to achieve our goal and overcome these challenges we propose a model based pose estimation method that tracks fast and notably occluded handed motions. We use a new abstract representation for the hand model and input data (Section 4) and define pose estimation as an optimization problem. We propose a novel objective function to maximize the similarity between the input depth data and the hand model, minimizing matching errors, and considering only biomechanically plausible poses (Section 5).

We evaluate our method against other competing methods on publicly available datasets (Section 6). We present several example interaction applications enabled by our method (Section 7). Our method does not require extrinsic calibration information and therefore supports a variety of close-range viewpoints, and even motion of the camera during acquisition. To demonstrate this, we show examples of egocentric interaction where the camera is mounted on the user's head.

4 Input and Model Representation

In order to perform fast and efficient pose estimation, compact representations of input data and hand models are essential. In the past, primitive shapes such as spheres or cylinders have been used to represent the hand [Oikonomidis et al. 2011b]. Similarly, downsampled images [Wang and Popovic 2009; Wang et al. 2011] or silhouettes [Ballan et al. 2012] have been used as representations

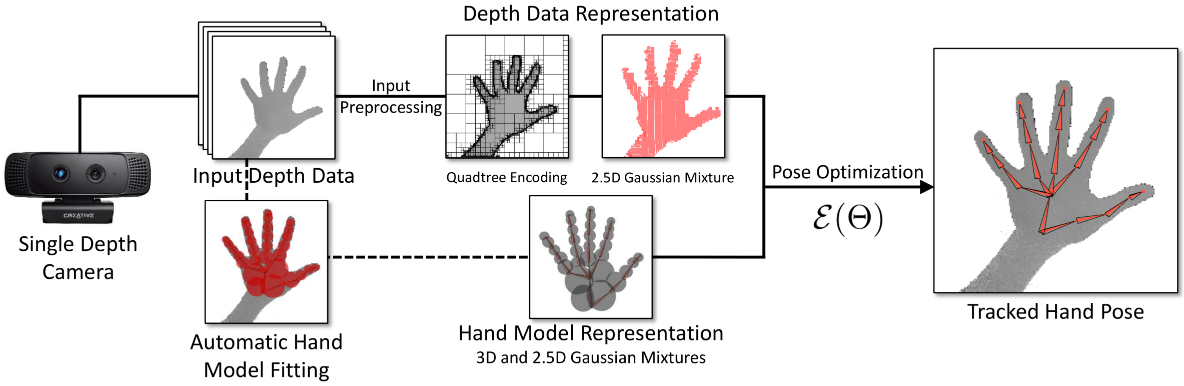

Figure 2: *Overview of our tracking pipeline. We automatically fit a user specific hand model to each person before tracking commences. This model is then used together with the Gaussian mixture representation of the depth data to perform pose optimization in real-time.*

of input data. Such compact representations allow fast optimization of model-to-image alignment, allow for efficient indexing into pose databases, often implicitly remove noise and other artifacts, and enable formulation of optimization problems.

Inspired by [Stoll et al. 2011], we use a collection of *weighted* Gaussian functions to represent both the input data and the hand model. Unlike their work, which uses multiple 2D color images and model-to-image alignment with a 2D error metric for pose optimization, we use a 2.5D formulation based on model alignment to a single depth image. An instance of the input depth or the hand model can be represented as a mixture of Gaussian functions

$$\mathcal{C}(\mathbf{x}) = \sum_{i=1}^{n} w_i \, \mathcal{G}_i(\mathbf{x}; \sigma, \boldsymbol{\mu}), \qquad (1)$$

where $\mathcal{G}_i(.)$ denotes a un-normalized Gaussian function with isotropic variance, σ^2, in all dimensions of $\mathbf{x} \in \mathcal{R}^n$, mean $\boldsymbol{\mu}$, and can be written as

$$\mathcal{G}_i(\mathbf{x}; \sigma, \boldsymbol{\mu}) := \exp\left[\frac{||\mathbf{x} - \boldsymbol{\mu}||}{2\sigma^2}\right]. \qquad (2)$$

The Gaussian mixture representation has many advantages. First, it enables our objective function to remain mathematically smooth, which allows analytic gradients to be computed. Second, only a few Gaussians are needed for representing the input depth and the hand model, which makes optimization fast. Finally, our Gaussian formulation provides a natural way to compute collisions within the context of an analytically differentiable objective function. Collision handling forms an important part of our objective function (Section 5). To aid visualization we represent each Gaussian in the mixture as a sphere ($\mathbf{x} \in R^3$) or circle ($\mathbf{x} \in R^2$) whose surface is the isosurface at standard deviation $1\,\sigma$. However, Gaussians have infinite support ($\mathcal{C}(\mathbf{x}) > 0$ everywhere) and can thus produce an attractive or repulsive *force* during optimization.

4.1 Depth Data Representation

The input to our method is in the form of a depth map where each pixel on the image has an associated depth value. In this data, only the surface facing the camera is visible and information about occluded regions is unavailable. We therefore propose a representation for only the front facing surface using Gaussian mixtures. We do this by first clustering the input based on depth and assigning each clustered region as a Gaussian.

First, we encode the depth image using quadtrees on the depth pixel grid where each node represents a homogeneous region of depth. We progressively downsample (by decimation) the orginal depth image to build an image pyramid. We then grow the quadtree by adding nodes for each part of the image where the difference in depth between the furthest and nearest points is below a threshold ϵ_c. In all our experiments, we set $\epsilon_c = 20$ mm.

The next step in our depth representation is to convert the quadtree into a suitable Gaussian mixture of the form in Equation 1. For each quad in the tree, we create a Gaussian function with $\boldsymbol{\mu}$ set to the center of the quad, and $\sigma = a/\sqrt{2}$, where a is the side of the quad. We also set each Gaussian function to have unit weight w_i since we consider all input data to be equally important. This leads us to an analytic representation of the *front facing surface* of the input depth, $\mathcal{C}_I(\mathbf{x}) = \sum_{q=1}^{n} \mathcal{G}_q(\mathbf{x})$, where $\mathbf{x} \in \mathcal{R}^2$ and n is the number of leaves in the quadtree. In addition, each quad has an associated depth value, d_q, which is the mean of all depth pixels within the quad. Figure 2 illustrates the process of converting input depth to a Gaussian mixture.

4.2 Hand Model

Given the analytic representation of the input depth, we need an equivalent representation of the hand so that we can formulate a measure of similarity between them. To this end, we model the volumetric extent of the hand using a collection of 3D Gaussian functions, $\mathcal{C}_h(\mathbf{x}) = \sum_{h=1}^{m} w_h \, \mathcal{G}_h(\mathbf{x})$ where $\mathbf{x} \in \mathcal{R}^3$ and m is the number of Gaussians. We assume that the best fitting model has Gaussians whose isosurface at $1\,\sigma$ coincides with the surface of the hand. Therefore, a new model of the hand needs to be constructed for each user. In Section 5.4 we present a fully automatic procedure to fit a hand model to a user.

Additionally, $\mathcal{C}_h$, is attached to a *parametric*, kinematic skeleton similar to that of [Simo Serra 2011], to enable movement of the Gaussians together with the skeleton joints. The skeleton is parametrized by pose parameters, $\Theta = \{\theta_j\}$ consisting of translational and angular components. We use $|\Theta| = 26$ parameters in our model consisting of 3 translational DOFs, 3 global rotations, and 21 joint angles. Thus, our goal is to find the best parameters Θ to match the input depth data. We also constrain the motion of joints by penalizing motions beyond plausible angle ranges (see Section 5.2).

Model Surface Representation The representation of the volumetric extent of the hand using 3D Gaussians cannot directly be used to optimize for the similarity to the input depth data, $\mathcal{C}_I$. This is because, $\mathcal{C}_I$ is a representation of the *front facing surface* while $\mathcal{C}_h$ represents the full volumetric extent of the hand. We therefore create an equivalent representation of the hand model that includes only the front facing parts.

For each Gaussian in $\mathcal{C}_h$, we create a new projected hand model, $\mathcal{C}_p = \sum_{p=1}^{m} w_p \, \mathcal{G}_p(\mathbf{x})$ where $\mathbf{x} \in R^2$ and $w_p = w_h \, \forall \, h$. $\mathcal{C}_p$ is a representation of the hand model as seen from the perspective of the depth camera and is defined over the depth image domain. The parameters of each Gaussian $\mathcal{G}_p$ are set to be $(\boldsymbol{\mu}_p, \sigma_p)$, where

$\boldsymbol{\mu}_p = \mathbf{K}\,[\,\mathbf{I}\,|\,\mathbf{0}\,]\,\boldsymbol{\mu}_h$. Like [Stoll et al. 2011] we approximate the perspective projection of a sphere (denoting an isotropic Gaussian) as a circle with a variance $\sigma_p = \sigma_h\, f / \left[\boldsymbol{\mu}_p\right]_z$. Here f is the focal length of the camera, and $\left[\boldsymbol{\mu}_p\right]_z$ denotes the z-coordinate of the Gaussian mean. This projected Gaussian mixture enables direct comparison of the depth data to the hand model as explained in Section 5.

5 Realtime Hand Tracking

In this section we describe our formulation of hand pose estimation as an optimization problem. We describe our objective function based on the Gaussian mixture representation and our procedure for optimization. The important advantage of our formulation is that the objective function is continuous and therefore its analytic gradient can be evaluated. This allows efficient optimization using simple and fast hill climbing methods. We also present an automatic method for obtaining a user specific hand model based on a greedy algorithm built on top of our objective function. Figure 2 gives an overview of our tracking pipeline.

5.1 Input Data Preprocessing

The first step in our tracking pipeline is preprocessing of the input depth data to obtain a Gaussian mixture model as in Equation 1. We first filter the input based the depth value such that pixels lying outside of an expected interaction range are removed. Because we assumed a close range interaction space, we set the near and far depths of the interaction range to be 150 mm and 600 mm. In our experiments we used a short range time of flight sensor which produces a noise commonly known as *flying pixels*. We apply a median filter which has been shown to be effective in reducing this kind of noise [Lefloch et al. 2013].

Finally, after the initial preprocessing steps we use the previously describe quadtree clustering method to create a Gaussian mixture representing the input, $\mathcal{C}_I$. Simultaneously, we create both 3D and 2.5D Gaussian mixtures for the hand model which are denoted by $\mathcal{C}_h$ and $\mathcal{C}_p$ respectively. Our optimization method works entirely on these Gaussian mixtures making efficient, fast optimization possible.

5.2 Objective Function

Our goal is to optimize for the skeleton pose parameters Θ that best explain the input data while accounting only for biomechanically plausible poses. We frame an objective function that satisfies our goal and yet remains mathematically smooth and suited for fast optimization. Our objective function is given as

$$\begin{aligned} \mathcal{E}(\Theta) = E_{sim} - w_c\, E_{col} - w_d\, E_{dan} \\ -w_l\, E_{lim} - w_s\, E_{smo}, \end{aligned} \tag{3}$$

where E_{sim} is a measure of similarity between $\mathcal{C}_I$ and $\mathcal{C}_p$, E_{col} is a penalty for collisions between Gaussians in $\mathcal{C}_h$, E_{dan} is a penalty for *dangling* parts of the model $\mathcal{C}_h$, E_{lim} enforces a soft constraint on the skeleton joint limits, E_{smo} enforces smoothness in the tracked motion. In all our experiments, the weighting factors for these different terms were set to the following: $w_c = 1.0$, $w_d = 0.1$, $w_l = 0.2$, and $w_s = 1.0$. Before describing each of the terms in detail we first introduce a measure of similarity between two Gaussian mixtures which is the basis for many of the terms in the obejctive.

Gaussian Similarity Measure We define a similarity measure between any two pairs of Gaussian mixtures $\mathcal{C}_a$ and $\mathcal{C}_b$ as,

$$\begin{aligned} E(\mathcal{C}_a, \mathcal{C}_b) &= \sum_{p \in \mathcal{C}_a} \sum_{q \in \mathcal{C}_b} w_p\, w_q \int_{\Omega} \mathcal{G}_p(\mathbf{x})\, \mathcal{G}_q(\mathbf{x})\, \mathrm{d}\mathbf{x} \\ &= \sum_{p \in \mathcal{C}_a} \sum_{q \in \mathcal{C}_b} D_{pq} \end{aligned} \tag{4}$$

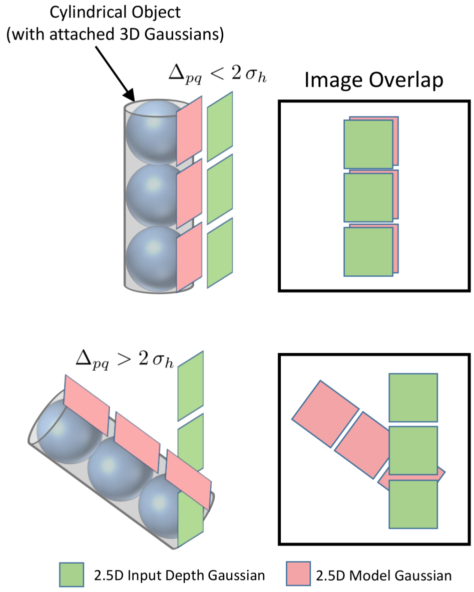
Cylindrical Object
(with attached 3D Gaussians)
$\Delta_{pq} < 2\,\sigma_h$
Image Overlap
$\Delta_{pq} > 2\,\sigma_h$
🟩 2.5D Input Depth Gaussian
🟥 2.5D Model Gaussian

Figure 3: *Consider the similarity value (E_{sim}) for a cylindrical shape represented by 3 Gaussians. The top figure shows a case where the value of E_{sim} is high since the image overlap is high and the depth difference Δ_{pq} is low. The bottom figure shows a case where the image overlap is moderate but $\Delta > 2\,\sigma_h$ thus making $E_{sim} = 0$.*

where Ω denotes the domain of integration of $\mathbf{x}$. The measure has a high value if the spatial support of the two Gaussian mixtures aligns well. This bears resemblance to the Bhattacharyya Coefficient used to measure the similarity of probability distributions while being computationally less extensive.

Similarity Term (E_{sim}) The similarity term measures the quality of overlap between the projected model Gaussian mixture $\mathcal{C}_p$ and the image Gaussian mixture $\mathcal{C}_I$. Additionally, this measure also incorporates the depth information available for each Gaussian in the mixture. Figure 3 explains this term intuitively. Two Gaussians that are close (in 2D pixel distance) in the depth image obtain a high value if their depth values are also close. On the other hand, the same Gaussians obtain a low value if their depths are too far apart. Formally, this term is defined as,

$$E_{sim}(\mathcal{C}_p, \mathcal{C}_I) = \frac{1}{E(\mathcal{C}_I, \mathcal{C}_I)} \sum_{p \in \mathcal{C}_p} \sum_{q \in \mathcal{C}_I} \Delta(p, q)\, D_{pq} \tag{5}$$

where D_{pq} is as defined in Equation 4 and

$$\Delta(p, q) = \begin{cases} 0, & \text{if } |d_p - d_q| \geq 2\,\sigma_h \\ 1 - \frac{|d_p - d_q|}{2\,\sigma_h}, & \text{if } |d_p - d_q| < 2\,\sigma_h. \end{cases}$$

Here, d_p and d_q are the depth values associated with each Gaussian in $\mathcal{C}_p$ and $\mathcal{C}_q$ respectively, and σ_h is the standard deviation of the *unprojected* model Gaussian $\mathcal{G}_h$. The depth value of each Gaussian in $\mathcal{C}_p$ is computed as $d_p = \left[\boldsymbol{\mu}_h\right]_z - \sigma_h$. The factor $E(\mathcal{C}_I, \mathcal{C}_I)$ is the overlap of the depth image with itself and serves to normalize the similarity term. The Δ factor has a support $[0, 1]$ thus ensuring the similarity between a projected model Gaussian and an image Gaussian is 0 if they lie too far apart in depth.

Collision Penalty (E_{col}) The fingers of a hand are capable of fast motions and often come in close proximity with one another causing aliasing of corresponding depth pixels in the input. Including a penalty for collisions avoids fingers *sticking* with one another. The 3D Gaussian mixture representation of the hand model ($\mathcal{C}_h$) offers an efficient way to penalize collisions because they implicitly act as collision proxies. We define the penalty for collisions as,

$$E_{col}(\Theta) = \frac{1}{E(\mathcal{C}_h, \mathcal{C}_h)} \sum_{p \in \mathcal{C}_h} \sum_{\substack{q \in \mathcal{C}_h \\ q > p}} D_{pq}, \tag{6}$$

where $E(\mathcal{C}_h, \mathcal{C}_h)$ is a normalization constant denoting the overlap of the hand model with itself. This term penalizes model Gaussians that collide with others but not if they collide with themselves. As we show in the results, the collision term has a large impact on the tracking performance.

Dangle Penalty Term (E_{dan}) The similarity term measures the quality of overlap between the projected model and the input data. However, it is a symmetric measure, *i.e.* the quality of overlap remains the same if $\mathcal{C}_I$ and $\mathcal{C}_p$ are inverted. This, together with the *repulsion* caused by the collision term, occasionally results in *dangling* fingers or parts of the hand model that do not explain any input data. We therefore add an additional term in our objective to penalize such poses.

Before we can penalize such poses we first detect parts of the model that are not explained by any input depth. We create a subset $\mathcal{D}$ of the Gaussians in C_p which are those Gaussians that are too far away from any depth input. Formally, this penalty is given as,

$$E_{dan}(\Theta) = \sum_{p \in \mathcal{D}} \sum_{q \in \mathcal{C}_I} \frac{\phi_{pq}}{D^0_{pq}} \left(D^0_{pq} - D_{pq}\right), \tag{7}$$

where

$$\phi(p, q) = \begin{cases} 0, & \text{if } ||\boldsymbol{\mu}_h - \boldsymbol{\mu}^b_q|| < \tau_1 \\ 0, & \text{if } ||\boldsymbol{\mu}_h - \boldsymbol{\mu}^b_q|| > \tau_2 \\ \frac{||\boldsymbol{\mu}_h - \boldsymbol{\mu}^b_q||}{(\tau_1 - \tau_2)}, & \text{otherwise.} \end{cases}$$

with τ_1 and τ_2 being the near and far thresholds to determine if a Gaussian is *dangling*. $\boldsymbol{\mu}_h$ is the 3D Gaussian corresponding to $\mathcal{G}_p$ and $\boldsymbol{\mu}^b_q$ is the back projected position of $\boldsymbol{\mu}_q$. The term D^0_{pq} denotes the overlap of two Gaussians with the same mean.

Joint Limit Penalty (E_{lim}) We add a penalty for poses that exceed predefined joint angle limits. This forces biomechanically plausible poses to be preferred over other poses. The joint limit penalty is given as,

$$E_{lim}(\Theta) = \sum_{\theta_j \in \Theta} \begin{cases} 0, & \text{if } \theta^l_j \le \theta_j \le \theta^h_j \\ ||\theta^l_j - \theta_j||^2, & \text{if } \theta_j < \theta^l_j \\ ||\theta_j - \theta^h_j||^2, & \text{if } \theta_j > \theta^h_j \end{cases} \tag{8}$$

where θ^l_j and θ^h_j are the lower and higher limits of the parameter θ_j which is defined based on anatomical studies of the hand [Simo Serra 2011].

Smoothness Penalty (E_{smo}) For fast hand motions optimization of pose parameters could create noise which manifests as jitter in tracking. To prevent this we penalize fast motions by adding a penalty as done by [Stoll et al. 2011]. This term is given as,

$$E_{smo}(\Theta_t) = \sum_{j \in \Theta_t} \left(0.5\ (\Theta_{t-2} + \Theta_t) - \Theta_{t-1}\right)^2 \tag{9}$$

where, Θ_t denotes the pose at time t. This terms acts as a regularizer and prevents jitter in the tracked pose.

5.3 Optimization

The goal of optimization is to find the pose Θ such that $\mathcal{E}(\Theta)$ is maximized. The objective function is well suited for gradient based optimization methods because we can derive the analytic gradient with respect to the degrees of freedom Θ. For efficiency, we adopt the fast gradient-based optimizer with adaptive step length proposed by [Stoll et al. 2011].

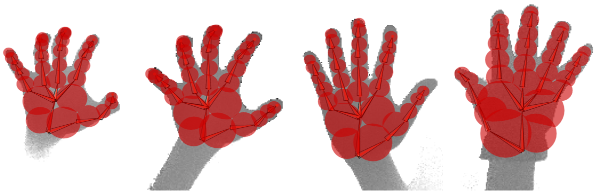

Figure 4: *Automatic fitting of user specific hand model for 4 subjects, one of whom is wearing a thick glove. The red spheres denote 3D Gaussians.*

For each input frame at time t we initialize the optimization with extrapolated parameters, $\Theta^0_t = \Theta_{t-1} + \alpha\, \Theta_{t-2}$. We experimented with parallel optimizations starting from multiple settings of α but found that a single optimization with $\alpha = 0.5$ worked best. For all realtime results we set the number of iterations to be 10.

5.4 User Specific Hand Modeling

Our pose optimization method works well when a customized hand model for each user is available. One way to create a user-specific model is to obtain a laser scan of the hand and manually assign the Gaussian mixture model. [Stoll et al. 2011] and [Sridhar et al. 2013] adopted a semi-automatic procedure where a known pose is used to optimize for shape and bone length parameters. Both these methods are time consuming and involve manual intervention.

In our experiments with different users we found that the primary variations in hand dimensions were finger thickness, hand length and width. We therefore opted for a simple strategy where a default skeleton and Gaussian mixture hand model is scaled using four parameters: hand length, width, depth, and variance of Gaussians. To find the specific scaling parameters for a user, we perform a greedy search over a fixed range for each scaling parameter. At each point on this parameter grid we evaluate the objective function value from Equation 3. The parameters that obtain the highest objective function value are selected as the model scaling parameters.

We found that this method works well for different users and can be easily done before tracking. This method is also fast and takes less than a second to find a user-specific hand model. Figure 4 shows some qualitative results from our model fitting strategy for different users.

6 Results and Evaluation

In this section we provide quantitative and qualitative evidence to show that our method performs well for fast motions and finger articulations. Evaluation of hand tracking algorithms is hard because of numerous reasons. First, obtaining ground truth information is difficult. Marker-based motion capture is often used for evaluating full-body tracking but these techniques do not work equally well for hands because of self-occlusions. Therefore, most methods have resorted to evaluation on synthetic data [Oikonomidis et al. 2011b; Oikonomidis et al. 2011a] which is not representative of real world hand motions. Second, there are no established benchmark datasets with accepted error metrics that can be used for relative comparison of different methods. Together with the unavailability of public implementations of methods, this makes relative comparison between methods difficult.

In this work, we evaluate our method on a publicly available dataset, Dexter 1 [Sridhar et al. 2013]. This dataset consists of fast challenging motions captured in a multiview setup including a close range depth camera; we only use the data from the latter sensour in our method. The sequences cover flexion–extension, abduction–adduction, and random fingerwaving motions. The fingertips are annotated manually in the depth data thus making it

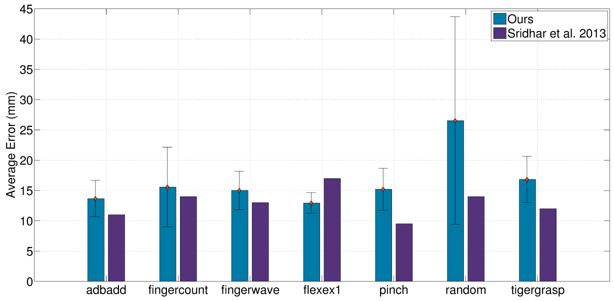

Figure 5: *Average error over the 7 sequences in Dexter 1 and comparison with the* ***multi-view*** *method of [Sridhar et al.]. Our single camera method runs at much higher frame rates and performs well in motions involving finger articulations (`flexex1`) but runs into a few errors for motions with global hand movement (`random`).*

suitable for us to compare with the multi-view approach of [Sridhar et al. 2013].

We also provide qualitative evidence of better tracking in comparison with [Melax et al. 2013]. Together both these comparisons show that our method does well on finger articulations. We also analyze the effect of different components of our objective function on the tracking accuracy and discuss limitations of our method.

6.1 Quantitative Evaluation

Average Fingertip Error Our first comparison is of the average fingertip localization error over each of the 7 sequences in the dataset. We use the same metric as [Oikonomidis et al. 2011a; Sridhar et al. 2013] to enable comparison of results. For each sequence, we compute the mean (Euclidean) error of the 5 fingertip positions averaged over all frames in a sequence.

Figure 5 shows our average errors compared with that of [Sridhar et al. 2013]. We achieve an average accuracy of 16.54 mm on this dataset which is highly competitive to the 13.24 mm obtained by [Sridhar et al. 2013], given that they use a multicamera setup that helps resolve occlusions. Our method uses only a single depth camera, does not need extrinsic camera calibration and yet runs many times faster than their method.

We observe that our method does particularly well for motions that involve fast articulation of fingers such as `flexex1` where we achieve a low error. Our worst performance was on the `random` sequence consisting of fast global rotation of the hand. One explanation for this could be the capture framerate of the sequence. Our method relies on faster cameras (60 fps) while the Dexter 1 sequences are captured only at 25 fps. We intend to address this issue in the future.

Error Frequency We also report the percentage of frames that have an error of less than x mm where $x \in \{15, 20, 25, 30\}$. This is a stricter measure of our performance and shows the type of motions where we do best. Tables 1 confirms the trend that our method performs well for finger articulations. In 5 out of the 7 sequences, our method results in tracking errors of less than 25 mm for more than 95% of the frames. A closer examination shows that all these sequences contain strong finger articulations.

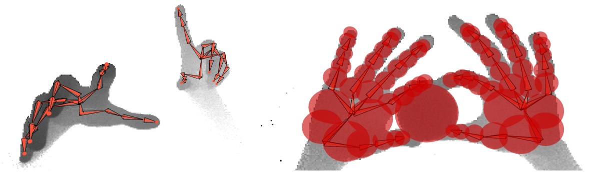

Figure 8: *Qualitative results from tracking two hands and an object being manipulated with two hands.*

Error < (mm)	adbadd	fingercount	fingerwave	pinch	random	tigergrasp	flexex1
15	79.3578	66.0377	56.6038	49.5238	29.1667	36.6972	88.4793
20	97.7064	92.4528	91.9811	89.0476	48.8095	87.1560	100.0000
25	98.6239	95.2830	99.0566	99.5238	58.9286	94.9541	100.0000
30	99.0826	95.7547	100.0000	100.0000	70.8333	98.1651	100.0000

Table 1: *Percentage of total frames in a sequence that have an error of less x mm.*

Effect of Objective Function Terms To further motivate the need for the different terms in our objective function we present a comparison of pose estimation by progressively disabling several terms. Figure 6 shows a plot of the average fingertip error over the `flexex1` sequence. Using only the similarity term E_{sim} results in catastrophic tracking failure. Adding the joint limits (E_{lim}) and smoothness (E_{smo}) terms helps recover from some of the failures but still produces unsatisfactory results. But the most significant performance gain is from the collision penalty (E_{col}) term. This confirms our own observations and previous work [Oikonomidis et al. 2011a] that collisions are an effective method to resolve self-occlusions, and we contribute with a very efficient and numerically advantageous way of testing for collisions during pose fitting. Adding the dangle penalty term (E_{dan}) leads to a further improvement in tracking accuracy with very few tracking errors.

6.2 Qualitative Results

Finally, we present several qualitative results of our method on realtime sequences in Figure 7. In the last row we also qualitatively compare our method to that of [Melax et al. 2013]. We can only do qualitative comparisons as their software does not allow to read data from disk, and works on the real-time stream from the camera. Therefore, we took care to reenact poses as closely as possible. We perform better than their method on motions with fast finger articulations such as pinching. However, we perform slightly worse on motions involving fast global rotation. We intend to explore solutions of this in future work.

The generality of our problem formulation allows us to track more than one hand. In Figure 8 we show some tracking results with two hands playing with a ball. While our framerate drops to 20 fps for multiple hands, this serves to demonstrate that our approach is extendable to more general cases.

Runtime Performance We obtained all our results on a machine with an Intel Xeon E5-1620 Processor and 16 GB of RAM (no GPU was used). We used the Intel Senz3D depth camera with a depth resolution of 320 × 240. The tracking results were optimized with 10 gradient ascent iterations. On the realtime sequences, image acquisition, preprocessing, and creating the Gaussian mixture representation took 2 ms. The optimization took between 18 and 20 ms for a framerate between 45 and 55 fps.

For our interaction applications we created a WebSocket connection through which tracking results were transmitted to the interactive applications. The transmission of data took a further 5 ms due to inter-process communication delays. However, this latency is quite low and does not adversely affect user experience.

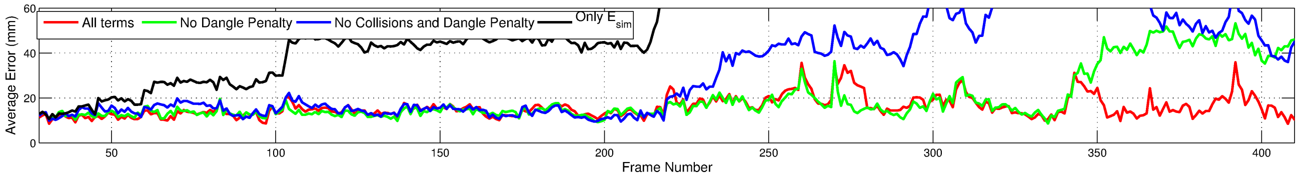

Figure 6: *Plot of the error for each term added to our objective function for the* `flexex1` *sequence. Using all the terms produces the lowest errors. In particular, the errors are consistently over* 50 *mm without the penalty for collisions. Best viewed in color.*

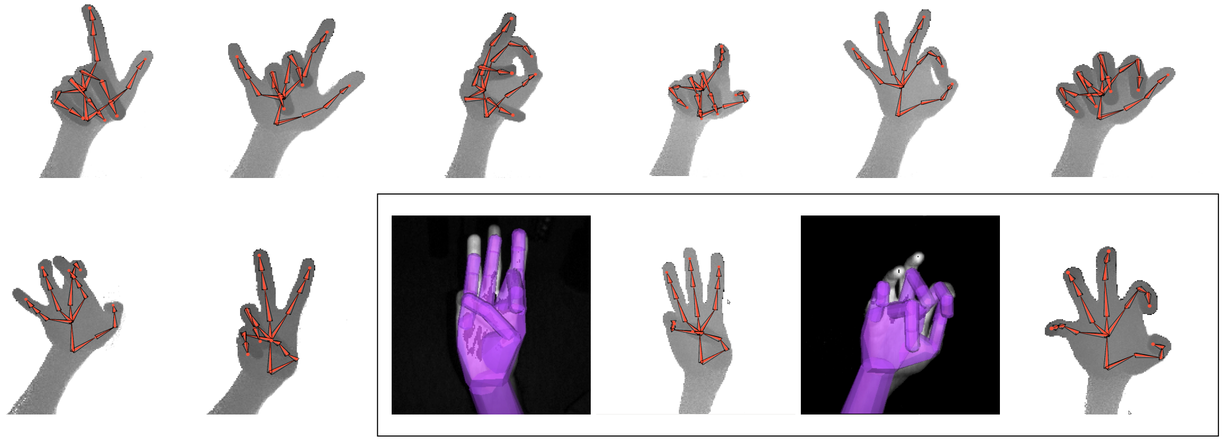

Figure 7: *Qualitative results from our tracking approach (top row and two leftmost in the bottom row). The highlighted box shows visual comparison with [Melax et al.]*

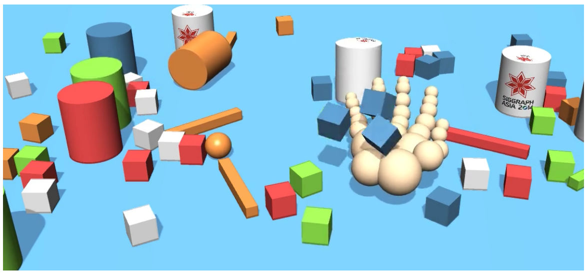

Figure 9: *Blocks world free hand interaction in action. Users can push and throw objects in the scene. Other actions can be mapped to pinch motions with different fingers.*

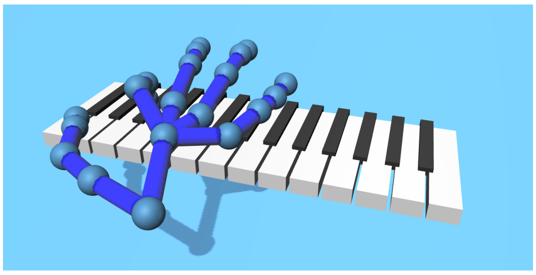

Figure 10: *Playing a virtual piano using finger articulations.*

7 Application Examples

We show several examples demonstrating the utility of our method for interactive applications. We specifically focus on 3D interactions that make use of fast finger articulations. Because our method is realtime and tracks all the DOFs of the hand we are able to enable more complex interactions.

We show three main applications. First, a virtual *blocks world* environment with physics where the user can add and manipulate virtual objects with fingers. Second, an example of playing a musical instrument in a 3D virtual environment with fingers. Finally, we show interaction in a mobile setting where the user's wearing a head-mounted camera.

Blocks World We created a virtual environment that resembles a table where a number of basic objects can be added and manipulated (Figure 9). This environment also contains realistic physics. We placed the camera at the bottom of the monitor in a typical desktop setting.

Users have three modes of interaction in this environment.

- Adding new *blocks* to the environment by a pinching gesture. Pinching with a different finger adds a different object (cubes, boxes, cylinders or spheres).
- Selecting objects and transforming their scale, position or rotation. This operation can be used to move objects around or arrange them in some order.
- Free-hand interaction with objects using a sphere representation of the hand to interact naturally with objects.

Previous work has shown how hand tracking [Wang et al. 2011] can be used for assembly tasks using just motion of hands and pinching with index finger. In our blocks world users can pinch with fingers, each of which is mapped to a particular interaction. Users can also move freely around the scene and make their finger articulations manipulate objects. With more gestures enabled by our method better interactions can be created.

Playing a Virtual Musical Instrument We also created an example of playing a virtual musical instrument using finger articulations. The depth camera was placed in a similar location as earlier in a desktop setting. Within the blocks world, we render a virtual piano with 14 keys. Each key is mapped to a single note.

To aid users, we render a skeleton representation of their hand as shown in Figure 10. Whenever the fingertip of the rendered skeleton hits a key, a note is played. More such musical instruments can

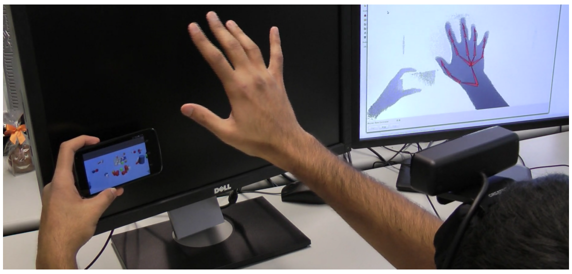

Figure 11: *Interaction in a mobile setting with a head-mounted camera.*

be incorporated in our environment. Together with haptic feedback, we envision that such virtual musical instruments may complement real instruments in the future.

Mobile Interaction Wearable and mobile devices are gaining more popularity among consumers. However, multi-touch remains the standard interaction modality for these devices. To demonstrate that full hand tracking can be used in such scenarios we mounted a depth camera on a user's head. The user is then able to interact with the blocks world with similar interaction techniques. Figure 11 shows and example of this kind of interaction. Please see the supplementary material for tracking results from this viewpoint.

8 Future Work

Our method is well suited for tracking fast articulated finger motion and is robust to self-occlusions. However, we have some difficulty with fast global rotations of the hand (please see supplementary video for examples). We intend to explore solutions to this problem by using parameter space transformations and better optimization techniques.

Currently, we require that the first frame for tracking be sufficiently close to a rest pose. This is a common way of initialization in tracking which is used, for example, in the Microsoft Kinect for full body tracking. We could, however, augment it with a detection strategy to make initialization more robust and failsafe.

We demonstrated tracking of one hand and two hands. A natural extension that we intend to explore are interactions of hands and complex real objects. We believe that strong formulation of the problem and addition of physics would help address the difficulty of this task.

9 Conclusion

In this paper, we presented a method for realtime tracking of hand and finger motion using a single depth camera. Our method is robust and tracks the hand at 50 fps without using a GPU. We contribute to the tracking literature by proposing a novel representation of the input data and hand model using a mixture of Gaussians. This representation allows us to formulate pose estimation as an optimization problem and efficiently optimize it using analytic derivatives. We evaluated our method on publicly available datasets and demonstrated the utility of our method on several interaction examples. We intend to make our method available as a software API so that interaction designers can develop new interactions using free hand motions.

References

Athitsos, V., and Sclaroff, S. 2003. Estimating 3D hand pose from a cluttered image. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings*, vol. 2, II – 432–9 vol.2.

Baak, A., Muller, M., Bharaj, G., Seidel, H.-P., and Theobalt, C. 2011. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *2011 IEEE International Conference on Computer Vision (ICCV)*, 1092 – 1099.

Ballan, L., Taneja, A., Gall, J., Van Gool, L., and Pollefeys, M. 2012. Motion capture of hands in action using discriminative salient points. In *Computer Vision ECCV 2012*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7577 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 640–653.

Bolt, R. A. 1980. Put-that-there: Voice and gesture at the graphics interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York, NY, USA, SIGGRAPH '80, 262270.

Erol, A., Bebis, G., Nicolescu, M., Boyle, R. D., and Twombly, X. 2007. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding 108*, 12 (Oct.), 52–73.

Ganapathi, V., Plagemann, C., Koller, D., and Thrun, S. 2012. Real-time human pose tracking from range data. In *Computer Vision ECCV 2012*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7577. Springer Berlin Heidelberg, Berlin, Heidelberg, 738–751.

Keskin, C., Kirac, F., Kara, Y., and Akarun, L. 2011. Real time hand pose estimation using depth sensors. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 1228 –1234.

Kurmankhojayev, D., Hasler, N., and Theobalt, C. 2013. Monocular pose capture with a depth camera using a sums-of-gaussians body model. In *Pattern Recognition*, J. Weickert, M. Hein, and B. Schiele, Eds., no. 8142 in Lecture Notes in Computer Science. Springer Berlin Heidelberg, Jan., 415–424.

Lefloch, D., Nair, R., Lenzen, F., Schfer, H., Streeter, L., Cree, M. J., Koch, R., and Kolb, A. 2013. Technical foundation and calibration methods for time-of-flight cameras. In *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, M. Grzegorzek, C. Theobalt, R. Koch, and A. Kolb, Eds., no. 8200 in Lecture Notes in Computer Science. Springer Berlin Heidelberg, Jan., 3–24.

Melax, S., Keselman, L., and Orsten, S. 2013. Dynamics based 3D skeletal hand tracking. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, I3D '13, 184184.

Oikonomidis, I., Kyriazis, N., and Argyros, A. 2011. Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *2011 IEEE International Conference on Computer Vision (ICCV)*, 2088–2095.

Oikonomidis, I., Kyriazis, N., and Argyros, A. 2011. Efficient model-based 3D tracking of hand articulations using kinect. British Machine Vision Association, 101.1–101.11.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. 2011. Real-time human pose recognition in parts from single depth images. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1297 –1304.

Simo Serra, E. 2011. *Kinematic Model of the Hand using Computer Vision*. PhD thesis, Institut de Robtica i Informtica Industrial.

Sridhar, S., Oulasvirta, A., and Theobalt, C. 2013. Interactive markerless articulated hand motion tracking using

RGB and depth data. In *2013 IEEE International Conference on Computer Vision (ICCV), (to appear)*.

STOLL, C., HASLER, N., GALL, J., SEIDEL, H., AND THEOBALT, C. 2011. Fast articulated motion tracking using a sums of gaussians body model. In *2011 IEEE International Conference on Computer Vision (ICCV)*, 951 –958.

STURMAN, D., AND ZELTZER, D. 1994. A survey of glove-based input. *IEEE Computer Graphics and Applications 14*, 1, 30–39.

TANG, D., YU, T.-H., AND KIM, T.-K. 2013. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *The IEEE International Conference on Computer Vision (ICCV)*.

WANG, R. Y., AND POPOVIC, J. 2009. Real-time hand-tracking with a color glove. *ACM Trans. Graph. 28*, 3 (July), 63:163:8.

WANG, R., PARIS, S., AND POPOVIC, J. 2011. 6D hands: markerless hand-tracking for computer aided design. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ACM, New York, NY, USA, UIST '11, 549558.

WANG, Y., MIN, J., ZHANG, J., LIU, Y., XU, F., DAI, Q., AND CHAI, J. 2013. Video-based hand manipulation capture through composite motion control. *ACM Trans. Graph. 32*, 4 (July), 43:143:14.

XU, C., AND CHENG, L. 2013. Efficient hand pose estimation from a single depth image. In *2013 IEEE International Conference on Computer Vision (ICCV)*, 3456–3462.

ZHAO, W., ZHANG, J., MIN, J., AND CHAI, J. 2013. Robust realtime physics-based motion control for human grasping. *ACM Trans. Graph. 32*, 6 (Nov.), 207:1207:12.

ZIMMERMAN, T. G., LANIER, J., BLANCHARD, C., BRYSON, S., AND HARVILL, Y. 1986. A hand gesture interface device. *SIGCHI Bull. 17*, SI (May), 189192.

Below you find a list of the most recent research reports of the Max-Planck-Institut für Informatik. Most of them are accessible via WWW using the URL `http://www.mpi-inf.mpg.de/reports`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
– Library and Publications –
Campus E 1 4

D-66123 Saarbrücken

E-mail: `library@mpi-inf.mpg.de`

MPI-I-2014-5-002	A. Anand, I. Mele, S. Bedathur, K. Berberich	Phrase Query Optimization on Inverted Indexes
MPI-I-2014-5-001	M. Dylla, M. Theobald	Learning Tuple Probabilities in Probabilistic Databases
MPI-I-2013-RG1-002	P. Baumgartner, U. Waldmann	Hierarchic superposition with weak abstraction
MPI-I-2013-5-002	F. Makari, R. Gemulla, R. Khandekar, J. Mestre, M. Sozio	A distributed algorithm for large-scale generalized matching
MPI-I-2013-1-001	S. Ott	New results for non-preemptive speed scaling
MPI-I-2012-RG1-002	A. Fietzke, E. Kruglov, C. Weidenbach	Automatic generation of inductive invariants by SUP(LA)
MPI-I-2012-RG1-001	M. Suda, C. Weidenbach	Labelled superposition for PLTL
MPI-I-2012-5-004	F. Alvanaki, S. Michel, A. Stupar	Building and maintaining halls of fame over a database
MPI-I-2012-5-003	K. Berberich, S. Bedathur	Computing n-gram statistics in MapReduce
MPI-I-2012-5-002	M. Dylla, I. Miliaraki, M. Theobald	Top-k query processing in probabilistic databases with non-materialized views
MPI-I-2012-5-001	P. Miettinen, J. Vreeken	MDL4BMF: Minimum Description Length for Boolean Matrix Factorization
MPI-I-2012-4-001	J. Kerber, M. Bokeloh, M. Wand, H. Seidel	Symmetry detection in large scale city scans
MPI-I-2011-RG1-002	T. Lu, S. Merz, C. Weidenbach	Towards verification of the pastry protocol using TLA+
MPI-I-2011-5-002	B. Taneva, M. Kacimi, G. Weikum	Finding images of rare and ambiguous entities
MPI-I-2011-5-001	A. Anand, S. Bedathur, K. Berberich, R. Schenkel	Temporal index sharding for space-time efficiency in archive search
MPI-I-2011-4-005	A. Berner, O. Burghard, M. Wand, N.J. Mitra, R. Klein, H. Seidel	A morphable part model for shape manipulation
MPI-I-2011-4-003	J. Tompkin, K.I. Kim, J. Kautz, C. Theobalt	Videoscapes: exploring unstructured video collections
MPI-I-2011-4-002	K.I. Kim, Y. Kwon, J.H. Kim, C. Theobalt	Efficient learning-based image enhancement : application to compression artifact removal and super-resolution
MPI-I-2011-4-001	M. Granados, J. Tompkin, K. In Kim, O. Grau, J. Kautz, C. Theobalt	How not to be seen inpainting dynamic objects in crowded scenes
MPI-I-2010-RG1-001	M. Suda, C. Weidenbach, P. Wischnewski	On the saturation of YAGO
MPI-I-2010-5-008	S. Elbassuoni, M. Ramanath, G. Weikum	Query relaxation for entity-relationship search
MPI-I-2010-5-007	J. Hoffart, F.M. Suchanek, K. Berberich, G. Weikum	YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia
MPI-I-2010-5-006	A. Broschart, R. Schenkel	Real-time text queries with tunable term pair indexes
MPI-I-2010-5-005	S. Seufert, S. Bedathur, J. Mestre, G. Weikum	Bonsai: Growing Interesting Small Trees
MPI-I-2010-5-004	N. Preda, F. Suchanek, W. Yuan, G. Weikum	Query evaluation with asymmetric web services

MPI-I-2010-5-003	A. Anand, S. Bedathur, K. Berberich, R. Schenkel	Efficient temporal keyword queries over versioned text
MPI-I-2010-5-002	M. Theobald, M. Sozio, F. Suchanek, N. Nakashole	URDF: Efficient Reasoning in Uncertain RDF Knowledge Bases with Soft and Hard Rules
MPI-I-2010-5-001	K. Berberich, S. Bedathur, O. Alonso, G. Weikum	A language modeling approach for temporal information needs
MPI-I-2010-1-001	C. Huang, T. Kavitha	Maximum cardinality popular matchings in strict two-sided preference lists
MPI-I-2009-RG1-005	M. Horbach, C. Weidenbach	Superposition for fixed domains
MPI-I-2009-RG1-004	M. Horbach, C. Weidenbach	Decidability results for saturation-based model building
MPI-I-2009-RG1-002	P. Wischnewski, C. Weidenbach	Contextual rewriting
MPI-I-2009-RG1-001	M. Horbach, C. Weidenbach	Deciding the inductive validity of $\forall\exists^*$ queries
MPI-I-2009-5-007	G. Kasneci, G. Weikum, S. Elbassuoni	MING: Mining Informative Entity-Relationship Subgraphs
MPI-I-2009-5-006	S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, G. Weikum	Scalable phrase mining for ad-hoc text analytics
MPI-I-2009-5-005	G. de Melo, G. Weikum	Towards a Universal Wordnet by learning from combined evidenc
MPI-I-2009-5-004	N. Preda, F.M. Suchanek, G. Kasneci, T. Neumann, G. Weikum	Coupling knowledge bases and web services for active knowledge
MPI-I-2009-5-003	T. Neumann, G. Weikum	The RDF-3X engine for scalable management of RDF data
MPI-I-2009-5-003	T. Neumann, G. Weikum	The RDF-3X engine for scalable management of RDF data
MPI-I-2009-5-002	M. Ramanath, K.S. Kumar, G. Ifrim	Generating concise and readable summaries of XML documents
MPI-I-2009-4-006	C. Stoll	Optical reconstruction of detailed animatable human body models
MPI-I-2009-4-005	A. Berner, M. Bokeloh, M. Wand, A. Schilling, H. Seidel	Generalized intrinsic symmetry detection
MPI-I-2009-4-004	V. Havran, J. Zajac, J. Drahokoupil, H. Seidel	MPI Informatics building model as data for your research
MPI-I-2009-4-003	M. Fuchs, T. Chen, O. Wang, R. Raskar, H.P.A. Lensch, H. Seidel	A shaped temporal filter camera
MPI-I-2009-4-002	A. Tevs, M. Wand, I. Ihrke, H. Seidel	A Bayesian approach to manifold topology reconstruction
MPI-I-2009-4-001	M.B. Hullin, B. Ajdin, J. Hanika, H. Seidel, J. Kautz, H.P.A. Lensch	Acquisition and analysis of bispectral bidirectional reflectance distribution functions
MPI-I-2008-RG1-001	A. Fietzke, C. Weidenbach	Labelled splitting
MPI-I-2008-5-004	F. Suchanek, M. Sozio, G. Weikum	SOFIE: a self-organizing framework for information extraction
MPI-I-2008-5-003	G. de Melo, F.M. Suchanek, A. Pease	Integrating Yago into the suggested upper merged ontology
MPI-I-2008-5-002	T. Neumann, G. Moerkotte	Single phase construction of optimal DAG-structured QEPs
MPI-I-2008-5-001	G. Kasneci, M. Ramanath, M. Sozio, F.M. Suchanek, G. Weikum	STAR: Steiner tree approximation in relationship-graphs
MPI-I-2008-4-003	T. Schultz, H. Theisel, H. Seidel	Crease surfaces: from theory to extraction and application to diffusion tensor MRI
MPI-I-2008-4-002	D. Wang, A. Belyaev, W. Saleem, H. Seidel	Estimating complexity of 3D shapes using view similarity
MPI-I-2008-1-001	D. Ajwani, I. Malinger, U. Meyer, S. Toledo	Characterizing the performance of Flash memory storage devices and its impact on algorithm design
MPI-I-2007-RG1-002	T. Hillenbrand, C. Weidenbach	Superposition for finite domains