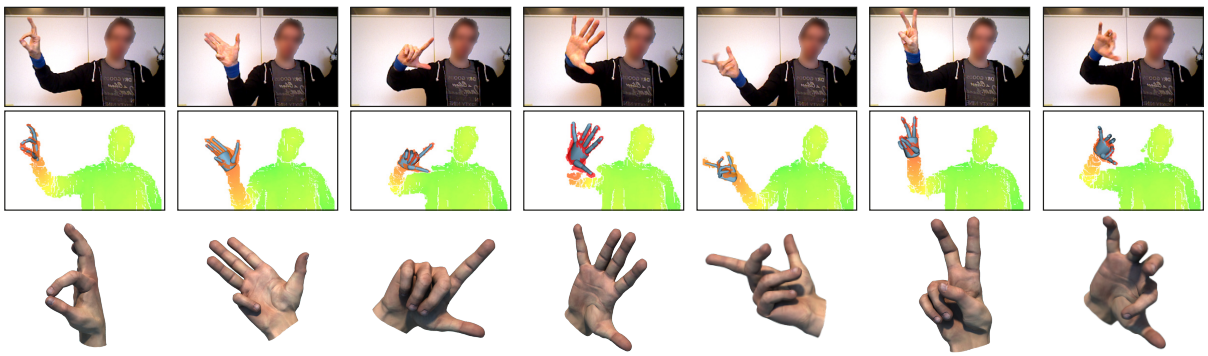
Eurographics Symposium on Geometry Processing 2015
Mirela Ben-Chen and Ligang Liu
(Guest Editors)

Volume 34 (2015), Number 5

Robust Articulated-ICP for Real-Time Hand Tracking

Andrea Tagliasacchi[1]* Matthias Schröder[2]* Anastasia Tkach[1] Sofien Bouaziz[1] Mario Botsch[2] Mark Pauly[1]

[1]École Polytechnique Fédérale de Lausanne (EPFL) [2]Bielefeld University (*equal contributors)



Abstract

We present a robust method for capturing articulated hand motions in realtime using a single depth camera. Our system is based on a realtime registration process that accurately reconstructs hand poses by fitting a 3D articulated hand model to depth images. We register the hand model using depth, silhouette, and temporal information. To effectively map low-quality depth maps to realistic hand poses, we regularize the registration with kinematic and temporal priors, as well as a data-driven prior built from a database of realistic hand poses. We present a principled way of integrating such priors into our registration optimization to enable robust tracking without severely restricting the freedom of motion. A core technical contribution is a new method for computing tracking correspondences that directly models occlusions typical of single-camera setups. To ensure reproducibility of our results and facilitate future research, we fully disclose the source code of our implementation.

1. Introduction

Tracking and animating humans in motion is a fundamental problem in computer graphics and computer vision. A particularly important question is how to accurately reconstruct the shape and articulation of human hands. Hand motion is a crucial component of non-verbal communication, plays an important role in the animation of humanoid avatars, and is central for numerous human-computer interfaces. Accurate realtime body tracking [SFC*11, WZC12] and face tracking [CHZ14] systems have been recently proposed. Hand tracking is now gaining traction in the research community as a next natural step towards a complete system for online human communication in desktop environments [OKA11a, MKO13, SRS*14, SMRB14, TSLP14]. Recent industrial trends in interaction systems for virtual environments have lead to the development of (closed source) software packages for the processing of RGBD data, like the Intel RealSense SDK, or purpose-designed hardware, like the Leap Motion and the Nimble sensors.

In this paper we introduce a system for *realtime* hand tracking suitable for personal desktop environments. Our *non-invasive* setup using a single commodity RGBD sensor does not require the user to wear a glove or markers. Such single-camera acquisition is particularly advantageous as it is cheap, does not require any sensor calibration, and does not impede user movements.

Accurate hand tracking with a non-invasive sensing device in realtime is a challenging scientific problem. Human hands are highly articulated and therefore require models with sufficiently many degrees of freedom to adequately describe the

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

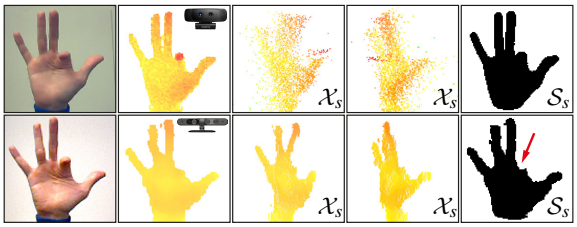


Figure 1: *The two different sensors used in our experiments provide data with substantially different characteristics. Top: Intel's Creative Interactive Gesture camera (time of flight) provides a complete silhouette image $\mathcal{S}_s$, but low quality depth measurements, resulting in severe noise in the point cloud $\mathcal{X}_s$. Bottom: Point clouds acquired by the PrimeSense camera (structured light) are much smoother, but the silhouette image can contain significant gaps.*

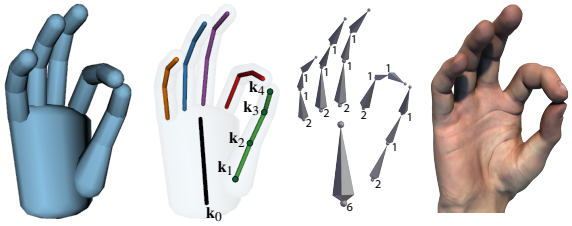


Figure 2: *A visualization of the template hand model with the number and location of degrees of freedom of our optimization. From left to right: The cylinder model used for tracking, the skeleton, the BVH skeleton exported to Maya to drive the rendering, the rendered hand model.*

corresponding motion space. Hand motion is often fast and exhibits intricate geometric configurations with complex contact patterns among fingers. With a single-camera RGBD setup, we are faced with incomplete data due to self-occlusions and high noise levels (see Figure 1). Yet the simplicity of the hardware and the ease of deployment make this setup the most promising for consumer applications as evidenced by the recent proliferation of new consumer-level sensors. To cope with the limited amount of available information, we employ an articulated template model as a geometric prior for shape completion and topology control. Our model does not only encode geometry, but also serves as a domain to represent information about *plausible* hand poses and motions. This statistical information, built by analyzing a database of annotated poses, is directly embedded into the optimization, which allows accurate tracking with a high number of degrees of freedom even in challenging scenarios.

Contributions. We present a complete system for realtime hand tracking using a single commodity RBGD input sensor. Our core technical contributions are:

- a novel articulated registration algorithm that efficiently integrates data and regularization priors into a *unified* real-time solver; see Section 4 and Appendix F,
- a combined 2D/3D registration method to align the 3D hand model to the acquired depth map and extracted silhouette image; see Section 4.1,
- a new way of computing data-to-model correspondences that accounts for occlusions and significantly improves the robustness of the tracking; see Section 4.1,
- a new regularization strategy that combines a statistical pose-space prior with kinematic and temporal priors to simultaneously ensure the inferred hand poses are plausible and aid the algorithm in recovering from loss-of-tracking; see Section 4.2,
- exposing an interesting relationship between the well known point-to-plane registration energy and Gauss-Newton linearization; see Appendix D.

Another important contribution of our paper is that we fully disclose our source code†. To the best of our knowledge, no other freely available implementation is available, and we believe that publishing our code will not only ensure reproducibility of our results, but also facilitate future research in this domain.

Note that there is a widespread belief [WZC12, ZSZ*14, QSW*14] that ICP-like techniques are too local and prone to local minima to successfully deal with fast articulated motion. One of our contributions is to show this commonly held belief should be re-considered. We demonstrate that a regularized geometric registration approach in the spirit of ICP can achieve outstanding performance. We believe this will significantly impact future research in this domain, as it will allow further development of registration techniques for real-time tracking, in contraposition to commonly employed techniques from the vision community like discriminative [TSLP14] and PSO [QSW*14] methods.

Our regularized geometric registration achieves robust, highly articulated hand tracking at up to 120 frames per second (fps). We quantitatively and qualitatively compare the performance of our algorithm to recent appearance-based and model-based techniques (see Section 6). These comparisons show a significant improvement in accuracy and robustness compared to the current state-of-the-art.

2. Related Work

We discuss the most relevant papers on human hand tracking related to our approach. For a more general overview of human motion analysis using depth sensors we refer to the recent survey of [YZW*13]. Tracking algorithms can be roughly divided into two main classes, *appearance-based* and *model-based* methods [EBN*07]. Appearance-based approaches train a classifier or a regressor to map image features to hand poses. Consequently, while these systems can robustly

† https://github.com/OpenGP/htrack

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

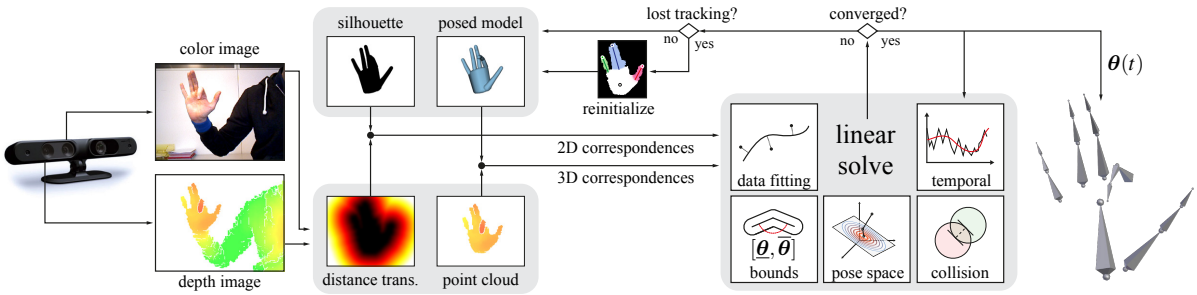


Figure 3: *Overview of our algorithm. For each acquired frame we extract a 3D point cloud of the hand and the 2D distance transform of its silhouette. From these we compute point correspondences to align a cylinder model of the hand to best match the data. This registration is performed in an ICP-like optimization that incorporates a number of regularizing priors to ensure accurate and robust tracking.*

determine a hand pose from a single frame, appearance-based methods are optimal in scenarios where only a rough pose estimate is desired [WPP11] or highly discriminative features can be extracted [WP09]. Conversely, *model-based* techniques approach tracking as an alignment optimization, where the objective function typically measures the discrepancy between the data synthesized from the model and the one observed by the sensor. While model-based methods can suffer from loss-of-tracking [WZC12], regularizing priors can be employed to infer high-quality tracking even when sensor data is incomplete or corrupted [MKO13, SMRB14]. In this work we focus on improving the robustness and accuracy of model-based approaches by combining effective 2D and 3D registration energies with carefully designed priors.

Appearance-based hand tracking. During the past few years, numerous appearance-based methods have been developed for hand tracking. Approaches based on nearest neighbor search [WP09, WPP11, RKEK13], decision trees [KKKA12, TYK13, TCTK14, KVK*14], or convolutional networks [TSLP14] have demonstrated that appearance-based methods can be successfully employed for realtime hand tracking. The strength of these methods is the capability of inferring a hand pose from a single frame without the need of relying on temporal coherence, which avoids drift. However, such appearance-based approaches are tightly linked to the training data and often do not generalize well to previously unseen hand poses, i.e., poses not contained in the training database. For this reason most of these methods assume a single hand in isolation to avoid data explosion, and often do not reach the accuracy of model-based methods.

Model-based hand tracking. A popular approach to hand motion capture is to use a marker-based system (e.g. Vicon, OptiTrack). A 3D hand model can then be fitted to the tracked markers to get the final hand poses. A small number of markers has been shown to be sufficient for reconstructing the 3D hand poses via inverse kinematics techniques [HRMO12]. However, due to frequent occlusions of the markers, motion sequences acquired using marker-based systems often need a significant amount of manual cleaning. To overcome this issue, [ZCX12] propose to complement a marker-based system with RGBD data to capture hand motion even in case of significant self-occlusion. Recently, accurate model-based tracking has been achieved in a multiple camera setup [SOT13, SRS*14], where the multiple vantage points help resolving challenging occlusions. Multiple camera systems have also been used successfully to model precise hand-hand and hand-object interactions [OKA11b, BTG*12, WMZ*13]. All of the above methods require a complex acquisition setup and manual calibration, which makes them less suitable for the kind of consumer-level applications that we target with this work.

Particle-swarm optimization (PSO) methods achieve interactive (15 fps) tracking with a single RGBD camera [OKA11a]. PSO techniques have also been applied successfully to model challenging interaction between two hands [OKA12] at a reduced rate of 4 fps. PSO is an optimization heuristic that does not use the gradient information of the considered optimization problem, but instead uses a sampling strategy. For this reason the accuracy and efficiency of PSO approaches heavily rely on the number of samples used. Oikonomidis et al. [OLA14] introduced a more advanced sampling strategy that improves tracking efficiency without compromising quality. However, gradient-based optimization approaches converge faster and more accurately than PSO when close to the solution, and are therefore well suited for realtime applications [QSW*14].

Compelling 60 fps realtime performance was recently shown using gradient-based optimization by [MKO13], where the optimization is expressed as a convex rigid body simulation, and numerous heuristics for re-initialization were employed to avoid tracking failures. Rather than resorting to reinitialization for robustness, [SMRB14] formulate the optimization in a subspace of likely hand poses. While the lower number of optimization variables leads to efficient computations, track-

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

ing accuracy can be limited by the reduced pose complexity induced by the subspace.

In this paper, we show that hand tracking can be formulated as a single gradient-based optimization to obtain an efficient and accurate real-time tracking system running at up to 120 fps. By using a combination of geometric and data-driven priors we achieve significant improvements in tracking quality and robustness.

3. Overview

Robust hand tracking with a commodity depth sensor is highly challenging due to self-occlusion, low quality/density of sensor data and the high degree of articulation of the human hand. We address these issues by proposing a regularized articulated ICP-like optimization that carefully balances data fitting with suitable priors (Figure 3). Our data fitting performs a joint *2D-3D* optimization. The 3D alignment ensures that every point measured by the sensor is sufficiently close to the tracked model $\mathcal{M}$. Simultaneously, as we cannot create such constraints for occluded parts of the hand, we integrate a 2D registration that pushes the tracked model to lie within the sensor visual hull. A carefully chosen set of priors regularizes the solution to ensure the recovered pose is plausible.

Acquisition device. Our system processes raw data acquired at 60 fps from a single RGBD sensor. Figure 1 illustrates this data for the *PrimeSense Carmine 1.09* structured light sensor as well as the *Creative Gesture Camera* time-of-flight sensor. From the raw data our algorithm extracts a 2D *silhouette image* $\mathcal{S}_s$ and a 3D *point cloud* $\mathcal{X}_s$. The two sensors exhibit different types of imperfections. The precision of depth measurements in the PrimeSense camera is significantly higher. However, substantial holes often occur at grazing angles, e.g. note the gap in the data where we would expect to see the index finger. Conversely, the Creative Gesture Camera provides an accurate and gap-free silhouette image, but suffers from high noise in the depth measurements, therefore resulting in very noisy point clouds. Our algorithm is designed to handle both types of imperfections. This is achieved by formulating an optimization that *jointly* considers silhouette and point cloud, balancing their contribution in a way that conforms to the quality of sensor data.

Tracking model. Our algorithm registers a template hand model to the sensor data. Similar to other techniques [OKA11a, SMRB14], we employ a simple (sphere capped) cylinder model as a geometric template; see Figure 2. We optimize for 26 degrees of freedom, 6 for global rotation and translation and 20 for articulation. Like in [MKO13], the model can be quickly adjusted to the user by specifying global scale, palm size and finger lengths. In most scenarios, it is sufficient to perform a simple uniform scaling of the model. Such a coarse geometry is sufficient for hand tracking, as the signal-to-noise ratio for commercially available RGBD

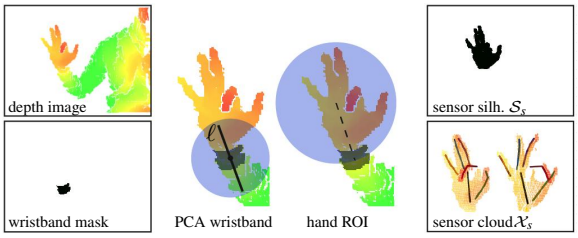


Figure 4: *We first identify the wristband mask by color segmentation, then compute the 3D orientation of the forearm as the PCA axis of points in its proximity. Offsetting a 3D sphere from the wristband center allows isolating the region of interest. The obtained silhouette image and sensor point clouds are shown on the right.*

sensors is low for samples on the fingers when compared to the size of a finger. Furthermore, the computation of closest-point correspondences can be performed in closed form and in parallel, which is essential for real-time performance. The hand's palm region may be better approximated by geometries other than a cylinder, but we found using only cylinder primitives to work well for tracking in terms of accuracy and efficiency. Furthermore, it simplified the implementation as the same correspondence computation routine can be used for all primitives in the model. While the geometry of the model used for tracking remains coarse, our algorithm computes joint angles (including rigid transformation) in the widespread *BVH* motion sequence format; these can be used to drive a high-resolution skinned hand rig as illustrated in Figure 2-d.

Preprocessing. The silhouette image $\mathcal{S}_s$ is not directly available from the sensor and needs to be computed. This labeling can be obtained by extracting the sensor color image and performing a skin color segmentation [OKA12,SMRB14], or can be obtained directly from depth images by performing a classification with randomized forests [TSLP14]. Another possibility is to exploit a full-body tracking algorithm [SFC*11] and segment the hand according to the wrist position. For gestural tracking, where the hand is typically the closest object to the sensor [QSW*14], a black wristband can be used to simplify segmentation by creating a gap in the depth image. Similarly to this method, in our system the user wears a *colored wristband*. We first identify the position of the wristband in the scene by color segmentation, then retrieve the 3D points in the proximity of the wristband and compute the principal axis. This axis, in conjunction with the wristband centroid, is then used to segment the hand point cloud. Any depth pixel within the hand point cloud is labelled as belonging to the silhouette image $\mathcal{S}_s$ as shown in Figure 4.

4. Optimization

In this section we derive the objective functions of our model-based optimization method and provide the rationales for

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

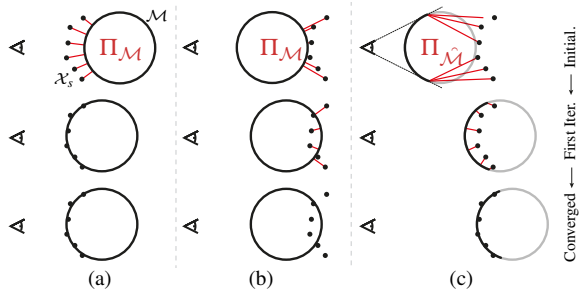


Figure 5: *Illustration of correspondences computations. The circles represent cross-sections of the fingers, the small black dots are samples of the depth map. (a) A configuration that can be handled by standard closest point correspondences. (b) Closest point correspondences to the back of the cylinder model can cause the registration to fall into a local minimum. Note that simply pruning correspondences with back-pointing normals would not solve this issue, as no constraints would remain to pull the finger towards the data. (c) This problem is resolved by taking visibility into account, and computing closest points only to the portion $\hat{\mathcal{M}}$ of $\mathcal{M}$ facing the camera.*

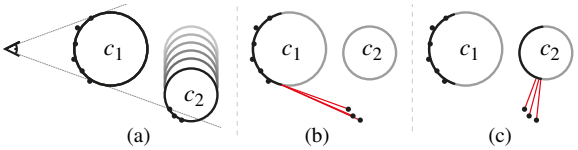


Figure 6: *Illustration of the impact of self-occlusion in correspondences computations. (a) The finger c_2 initially occluded by finger c_1 becomes visible, which causes new samples to appear. (b) Closest correspondences to the portion of the model visible from the camera do not generate any constraints that pull c_2 toward its data samples. This is the approach in [WZC12], where these erroneous matches are then simply pruned. (c) Our method also considers front-facing portions of the model that are occluded, allowing the geometry to correctly register.*

our design choices. Let $\mathcal{F}$ be the sensor input data consisting of a 3D point cloud $\mathcal{X}_s$ and 2D silhouette $\mathcal{S}_s$ (see Figure 1). Given a 3D hand model $\mathcal{M}$ with joint parameters $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_{26}\}$, we aim at recovering the pose $\boldsymbol{\theta}$ of the user's hand, matching the sensor input data $\mathcal{F}$. To achieve this goal, we solve the optimization problem

$$\min_{\boldsymbol{\theta}} \underbrace{E_{3\mathrm{D}} + E_{2\mathrm{D}} + E_{\mathrm{wrist}}}_{\text{Fitting terms}} + \underbrace{E_{\mathrm{pose}} + E_{\mathrm{kin.}} + E_{\mathrm{temporal}}}_{\text{Prior terms}}, \quad (1)$$

combining fitting terms that measure how well the hand parameters $\boldsymbol{\theta}$ represent the data frame $\mathcal{F}$, with prior terms that regularize the solution to ensure realistic hand poses. For brevity of notation we omit the arguments $\boldsymbol{\theta}, \mathcal{X}_s, \mathcal{S}_s$ of the energy terms. We first introduce the fitting terms and present our new solution to compute tracking correspondences. Then we discuss the prior terms and highlight their benefits in terms of tracking accuracy and robustness. More details on the implementation of the optimization algorithm will be given in Section 5 and the appendix.

4.1. Fitting Energies

Point cloud alignment. The term $E_{3\mathrm{D}}$ models a 3D geometric registration in the spirit of ICP as

$$E_{3\mathrm{D}} = \omega_1 \sum_{\mathbf{x} \in \mathcal{X}_s} \|\mathbf{x} - \Pi_{\mathcal{M}}(\mathbf{x}, \boldsymbol{\theta})\|_2, \quad (2)$$

where $\|\cdot\|_2$ denotes the ℓ_2 norm, $\mathbf{x}$ represents a 3D point of $\mathcal{X}_s$, and $\Pi_{\mathcal{M}}(\mathbf{x}, \boldsymbol{\theta})$ is the projection of $\mathbf{x}$ onto the hand model $\mathcal{M}$ with hand pose $\boldsymbol{\theta}$. Note that we compute a sum of absolute values of the registration residuals, not their squares. This corresponds to a mixed ℓ_2/ℓ_1 norm of the stacked vector of the residuals. For 3D registration such a sparsity-inducing norm has been shown to be more resilient to noisy point clouds containing a certain amount of outliers such as the ones produced by the Creative sensor (Figure 1). We refer to [BTP13] for more details.

3D correspondences. The 3D registration term involves computing the corresponding point $\mathbf{y} = \Pi_{\mathcal{M}}(\mathbf{x}, \boldsymbol{\theta})$ on the cylinder model $\mathcal{M}$ for each sensor point $\mathbf{x} \in \mathcal{X}_s$. In contrast to standard closest point search, we define the correspondence $\mathbf{y}$ as the closest point on the *front-facing* part $\hat{\mathcal{M}}$ of $\mathcal{M}$. This includes parts of the model that are oriented towards the camera but occluded by other parts. In our experiments we learned that this seemingly simple extension proved absolutely essential to obtain high-quality tracking results. Only considering model points that are visible from the sensor viewpoint, i.e., matching to the rendered model, is not sufficient for handling occlusions or instances of disappearing and reappearing sensor data; see Figure 5 and Figure 6.

To calculate $\mathbf{y}$, we first compute the closest points $\mathbf{x}_{\mathcal{C}}$ of $\mathbf{x}$ to each cylinder $\mathcal{C} \in \mathcal{M}$. Recall that our hand model consists of sphere-capped cylinders so these closest points can be

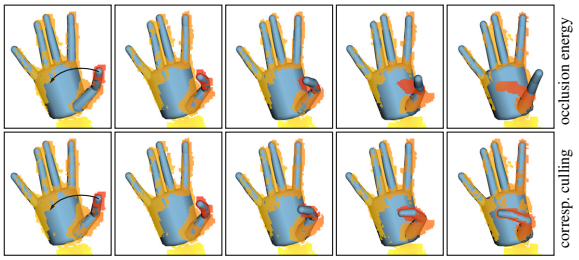


Figure 7: *Correspondence computations. The top row shows the strategy used in [QSW*14] adapted to our gradient-based framework according to the formulation given in [WZC12]. The bottom row shows the improved accuracy of our new approach.*

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

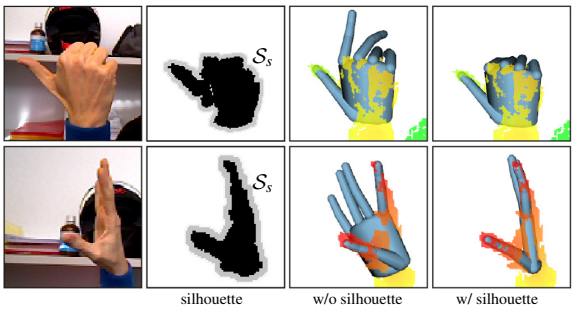


Figure 8: *Our 2D silhouette registration energy is essential to avoid tracking errors for occluded parts of the hand. When no depth data is available for certain parts of the model, a plausible pose is inferred by ensuring that the model is contained within the sensor silhouette image $\mathcal{S}_s$.*

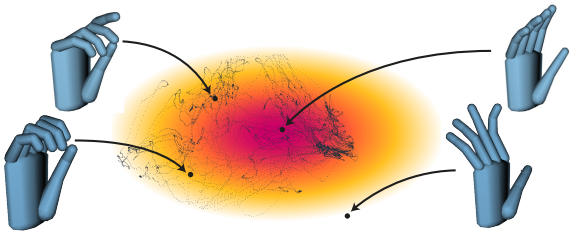


Figure 9: *An illustration of the PCA pose-space used to regularize the optimization. Black dots denote the samples of the data base. High likelihood poses are located nearby the mean of the latent space (dark red). The eigenvalues of the PCA define the metric in the low-dimensional space, skewing it in certain directions. Poses that, according to this metric, are far from the mean are likely to be unnatural and will be penalized in the optimization.*

computed efficiently in closed form and in parallel for each $\mathbf{x} \in \mathcal{X}_s$. We then identify back-facing points using the dot product of the cylinder surface normal $\mathbf{n}$ at $\mathbf{x}_\mathcal{C}$ and the view ray vector $\mathbf{v}$. For efficiency reasons, we use a simplified orthographic camera model where the view rays are constant, i.e., $\mathbf{v} = [0\ 0\ 1]^T$. If a point on a cylinder is back-facing ($\mathbf{n}^T\mathbf{v} > 0$), we project $\mathbf{x}$ onto the cylinder's silhouette contour line from the camera perspective, whose normals are orthogonal to $\mathbf{v}$.

A different strategy to address visibility issues has been introduced in [QSW*14]. These methods propose an energy that penalizes areas of the model falling in front of the data, which is then optimized using particle swarms. This energy can be integrated into our optimization following the formulation in [WZC12, Eq. 15]. However, such an energy is prone to create local minima in gradient-based optimization, as illustrated in Figure 7. Here the thumb has difficulty entering the palm region, as it must occlude palm samples before reaching its target configuration. Our correspondence search avoids such problems. Furthermore, note how [QSW*14] follows a *hypothesize-and-test* paradigm where visibility constraints in the form of *ray-casting* are easy to include. As discussed in [GPKT12], such constraints are much more difficult to include in iterative optimization techniques like ours. However, our front-facing correspondences computation provides a simple and elegant way to deal with such shortcomings.

Silhouette alignment. The 3D alignment energy $E_{3\mathrm{D}}$ robustly measures the distance between every point in the 3D point cloud $\mathcal{X}_s$ to the tracked model $\mathcal{M}$. However, as hands are highly articulated, significant self-occlusions are common during tracking. Such self-occlusions are challenging, because occluded parts will not be constrained when only using a 3D alignment energy. For this reason, we use a 2D silhouette term $E_{2\mathrm{D}}$ that models the alignment of the 2D silhouette of our rendered hand model with the 2D silhouette extracted from the sensor data as

$$E_{2\mathrm{D}} = \omega_2 \sum_{\mathbf{p}\in\mathcal{S}_r} \|\mathbf{p} - \Pi_{\mathcal{S}_s}(\mathbf{p},\boldsymbol{\theta})\|_2^2, \tag{3}$$

where $\mathbf{p}$ is a 2D point of the *rendered* silhouette $\mathcal{S}_r$, and $\Pi_{\mathcal{S}_s}(\mathbf{p},\boldsymbol{\theta})$ denotes the projection of $\mathbf{p}$ onto the *sensor* silhouette $\mathcal{S}_s$. Figure 8 shows why the silhouette term is crucial to avoid erroneous poses when parts of the model are occluded. Without the silhouette energy, the occluded fingers can mistakenly move to wrong locations, since they are not constrained by any samples in the depth map.

2D correspondences. In Equation 3, we compute the silhouette image $\mathcal{S}_r$ by first rendering the hand model $\mathcal{M}$ from the viewpoint of the sensor, caching the bone identifier and the 3D location associated with each pixel in a texture. The projection function $\Pi_{\mathcal{S}_s}(\mathbf{p},\boldsymbol{\theta})$ to compute the closest corresponding point to the sensor silhouette is evaluated efficiently using the 2D distance transform of $\mathcal{S}_s$. We use the linear time algorithm of [FH12] and store at every pixel the index to the closest correspondence.

Wrist alignment. The inclusion of the forearm for hand tracking has been shown beneficial in [MKO13]. Our wrist alignment energy encodes a much simplified notion of the forearm in the optimization that enforces the wrist joint to be located along its axis.

$$E_{\mathrm{wrist}} = \omega_3 \|\Pi_{2\mathrm{D}}(\mathbf{k}_0(\boldsymbol{\theta})) - \Pi_\ell(\mathbf{k}_0(\boldsymbol{\theta}))\|_2^2, \tag{4}$$

Minimizing this energy helps preventing the hand from erroneously rotating/flipping during tracking; an occurrence of this can be observed at 04:03 in the accompanying video. Here $\mathbf{k}_0$ is the 3D position of the wrist joint, and ℓ is the 2D line extracted by PCA of the 3D points associated with the wristband; see Figure 4. Note that $\Pi_{2\mathrm{D}}$ causes residuals to be minimized in screen-space, therefore the optimization of this energy will be analogous to the one of Equation 3. We

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

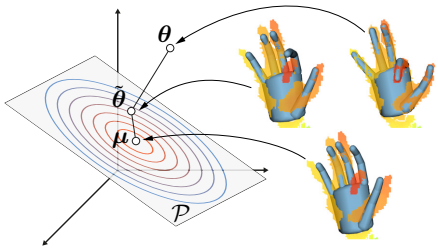


Figure 10: *An illustration of the energies involved in our pose-space prior. For illustration purposes the full dimensional parameter vector $\boldsymbol{\theta} \in \mathbb{R}^3$, while latent space variable $\tilde{\boldsymbol{\theta}} \in \mathbb{R}^2$. The PCA optimization in [SMRB14] constrains the pose parameters $\boldsymbol{\theta}$ to lie on the subspace $\mathcal{P}$. Conversely, we penalize the distance of our pose from $\mathcal{P}$ (Equation 5); simultaneously, we ensure our pose remains likely by preventing it from diverging from the mean of the distribution (Equation 6).*

optimize in screen space because, due to occlusion, we are only able to observe half of the wrist and this causes its axis to be shifted toward the camera.

4.2. Prior Energies

Minimizing the fitting energies alone easily leads to unrealistic or unlikely hand poses, due to the deficiencies in the input data caused by noise, occlusions, or motion blur. We therefore regularize the registration with data-driven, kinematic, and temporal priors to ensure that the recovered hand poses

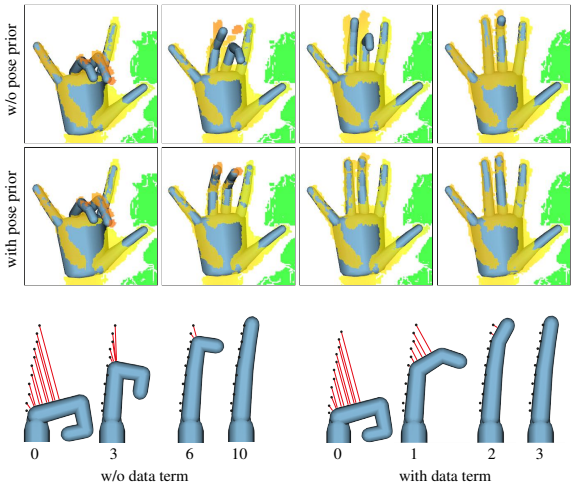


Figure 11: *Beyond favoring natural poses, the data prior term also positively affects convergence speed. Top: With the same number of iterations, only with activated data term does the model fully register to the scan. The illustration below shows how the same final state requires significantly fewer iterations with the data term.*

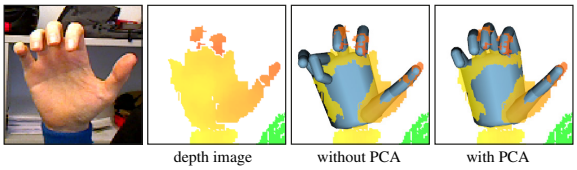


Figure 12: *Our pose-space regularization using a PCA prior ensures that a meaningful pose is recovered even when significant holes occur in the input data.*

are plausible. Each of these terms plays a fundamental role in the stability of our tracking algorithm, as we illustrate below.

Pose Space prior (data-driven). The complex and highly coupled articulation of human hands is difficult to model directly with geometric or physical constraints. Instead, we use a publicly available database of recorded hand poses [SMRB14] to create a data-driven prior E_{pose} that encodes this coupling. We construct a low-dimensional subspace of plausible poses by performing dimensionality reduction using PCA (see Figure 9). We enforce the hand parameters $\boldsymbol{\theta}$ to lie close to this low-dimensional linear subspace using a data term $E_{\text{pose}} = E_{\text{projection}} + E_{\text{mean}}$. To define the data term, we introduce auxiliary variables $\tilde{\boldsymbol{\theta}}$, i.e, the PCA weights, representing the (not necessarily orthogonal) projection of the hand pose $\boldsymbol{\theta}$ onto the subspace; see Figure 10. The projection energy measures the distance between the hand parameters and the linear subspace as

$$E_{\text{projection}} = \omega_4 \|(\boldsymbol{\theta} - \boldsymbol{\mu}) - \Pi_{\mathcal{P}} \tilde{\boldsymbol{\theta}}\|_2^2, \tag{5}$$

where $\boldsymbol{\mu}$ is the PCA mean. The matrix $\Pi_{\mathcal{P}}$, i.e., the PCA basis, reconstructs the hand posture from the low-dimensional space. To avoid unlikely hand poses in the subspace, we regularize the PCA weights $\tilde{\boldsymbol{\theta}}$ using an energy

$$E_{\text{mean}} = \omega_5 \|\boldsymbol{\Sigma} \tilde{\boldsymbol{\theta}}\|_2^2. \tag{6}$$

$\boldsymbol{\Sigma}$ is a diagonal matrix containing the inverse of the standard deviation of the PCA basis. Our tracking optimization is modified to consider the pose space by introducing the auxiliary variable $\tilde{\boldsymbol{\theta}}$ and then jointly minimizing over $\boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}$. The difference between our approach and optimizing directly in the subspace is further discussed in Appendix A. Note how the regularization energy in Equation 6 helps the tracking system recover from tracking failures. When no sensor constraints are imposed on the model, the optimization will attempt to push the pose toward the mean – a statistically likely pose from which tracking recovery is highly effective.

Figure 12 illustrates how the PCA data prior improves tracking by avoiding unlikely poses, in particular when the input data is incomplete. We found that even when data coverage is sufficient to recover the correct pose, the data term improves the convergence of the optimization as illustrated in Figure 11. Figure 13 shows how our regularized projective PCA formu-

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

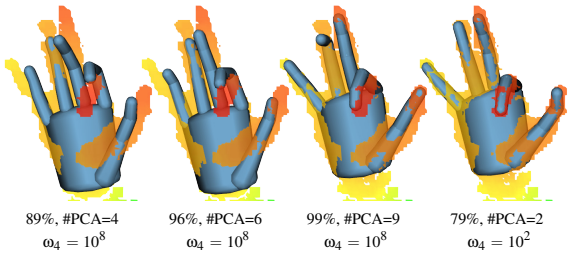


Figure 13: *Optimizing directly in the PCA subspace [SMRB14] can lead to inferior registration accuracy. We replicate this behavior by setting ω_4 in Equation 5 to a large value. Even when increasing the number of PCA bases to cover 99% of the variance in the database, the model remains too stiff to conform well to the input. Our approach is able to recover the correct hand pose by optimizing for projection distances even with a very limited number of bases (right).*

lation outperforms the direct subspace optimization proposed in previous work.

Kinematic prior. The PCA data term is a computationally efficient way of approximating the space of plausible hand poses. However, the PCA model alone cannot guarantee that the recovered pose is realistic. In particular, since the PCA is symmetric around the mean, fingers bending backwards beyond the physically realistic joint angle limits are not penalized by the data prior. Similarly, the PCA model is not descriptive enough to avoid self-intersections of fingers. These two aspects are addressed by the kinematic prior $E_{\text{kinematic}} = E_{\text{collision}} + E_{\text{bounds}}$. Under the simplifying assumption of a cylinder model, we can define an energy $E_{\text{collision}}$ that accounts for the inter-penetration between each pair of (sphere-capped) cylinders:

$$E_{\text{collision}} = \omega_6 \sum_{\{i,j\}} \chi(i,j)(d(\mathbf{c}_i,\mathbf{c}_j) - r)^2, \qquad (7)$$

where the function $d(\cdot,\cdot)$ measures the Euclidean distance between the cylinders axes $\mathbf{c}_i$ and $\mathbf{c}_j$, and r is the sum of the cylinder radii. $\chi(i,j)$ is an indicator function that evaluates to one if the cylinders i and j are colliding, and to zero otherwise. The top row of Figure 14 shows how this term avoids interpenetrations of the fingers.

To prevent the hand from reaching an impossible posture by overbending the joints, we limit the joint angles of the hand model:

$$E_{\text{bounds}} = \omega_7 \sum_{\theta_i \in \boldsymbol{\theta}} \underline{\chi}(i)(\theta_i - \underline{\theta}_i)^2 + \overline{\chi}(i)(\theta_i - \overline{\theta}_i)^2, \qquad (8)$$

where each hand joint is associated with conservative bounds $[\underline{\theta}_i, \overline{\theta}_i]$. For the bounds, we use the values experimentally determined by [CD95]. $\underline{\chi}(i)$ and $\overline{\chi}(i)$ are indicator functions. $\underline{\chi}(i)$ evaluates to one if $\theta_i < \underline{\theta}_i$, and to zero otherwise. $\overline{\chi}(i)$ is

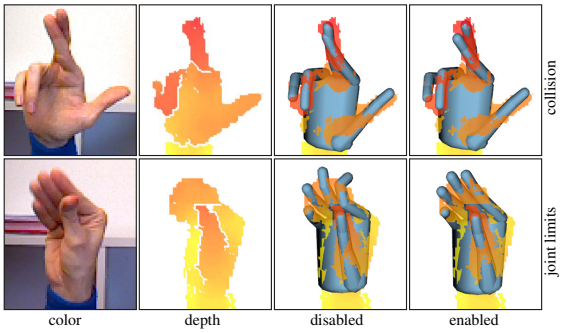


Figure 14: *Kinematic priors augment the data prior to account for inconsistencies in the pose space. The collision term avoids self-collisions (top row), while the term for joint angle bounds avoids overbending of the finger joints.*

equal to one if $\theta_i > \overline{\theta}_i$, and zero otherwise. The bottom row of Figure 14 illustrates the effect of the joint angle bounds.

Temporal prior. A common problem in particular with appearance-based methods are small-scale temporal oscillations that cause the tracked hand to jitter. A standard way to enforce temporal smoothness is to penalize the change of model parameters $\boldsymbol{\theta}$ through time, for example, by penalizing a quadratic energy accounting for velocity $\|\dot{\boldsymbol{\theta}}\|^2$ and acceleration $\|\ddot{\boldsymbol{\theta}}\|^2$ [WZC12]. However, if we consider a perturbation of the same magnitude, it would have a much greater effect if applied at the root, e.g., global rotation, than if applied to an element further down the kinematic tree, e.g., the last phalanx of a finger. Therefore, we propose a solution that measures the velocity and acceleration of a set of points attached to the

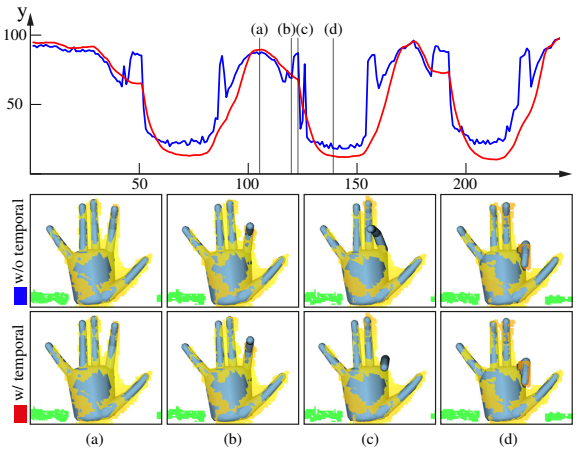


Figure 15: *The effect of the temporal prior. The graph shows the trajectory of the y-coordinate of the fingertip over time as the index finger is bend up and down repeatedly. The temporal prior reduces jitter, but also helps avoiding tracking artifacts that arise when fragments of data pop in and out of view.*

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

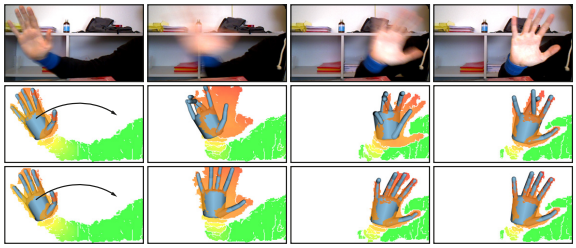


Figure 16: *During fast motion, optimizing directly for a fully articulated hand can lead to incorrect correspondences and cause loss of tracking (middle row). By compensating for the rigid motion ahead of solving for joint angles, our system can better capture fast movements (bottom row).*

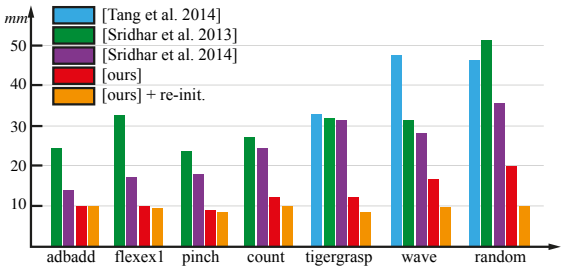


Figure 17: *We quantitatively evaluate our algorithm on the **Dexter-1** dataset from [SOT13]. The measurements report the root mean square errors of fingertip placements. The acquisition setup consists of several calibrated video cameras and a single depth camera. For our results and the method of [TCTK14], only the depth image is used for tracking, while the algorithms of Sridhar and colleagues also use the video streams. The blue, green, and purple bars are reproduced from [SRS*14]. For our algorithm we report results without (red) and with (orange) reinitialization.*

kinematic chain. We consider the motion of vertices $\mathbf{k}$ of the kinematic chain $\mathcal{K}$ (Figure 2) and build an energy penalizing the velocity and acceleration of these points:

$$E_{\text{temporal}} = \omega_8 \sum_{\mathbf{k}_i \in \mathcal{K}} \|\dot{\mathbf{k}}(\boldsymbol{\theta})\|_2^2 + \omega_9 \sum_{\mathbf{k}_i \in \mathcal{K}} \|\ddot{\mathbf{k}}(\boldsymbol{\theta})\|_2^2. \quad (9)$$

Figure 15 illustrates how the temporal prior reduces jitter and improves the overall robustness of the tracking; see also accompanying video.

5. Implementation

In this section we provide more details on the implementation of our optimization algorithm. The derivation of the necessary gradients and Jacobians is given in the appendix.

Optimization. The optimization of the tracking energy of Equation 1 over the pose $\boldsymbol{\theta}$ is performed by solving the non-linear least squares problem with a Levenberg-Marquardt approach. The assumption is that a current estimate of $\boldsymbol{\theta}$ is known from which we then compute an update. More specifically, the high acquisition speed of the sensing device allows us to employ the optimized parameters from the previous time frame as the starting estimate. We then iteratively approximate the energy terms using Taylor expansion and solve a linear system to get the update $\delta\boldsymbol{\theta}$ at each iteration (see appendix). As our algorithm achieves 60 fps tracking, the previously reconstructed pose is of sufficiently high quality allowing our solve to converge within seven iterations.

Initialization. As a user enters the scene our method is initialized by the fingertip detection and fitting from [QSW*14]. Other appearance-based methods could be used for initialization as well [TSLP14]. We also re-initialize the tracking in case a severe tracking failure is detected using the method of [WZC12]. Such re-initialization occurs rarely (e.g. less than 0.5% of the frames in the sequence of Figure 21).

Rigid bias. To improve the convergence of our solver in case of fast motion, we first perform the optimization in Equation 1 for the rigid motion only by optimizing for the root of the kinematic chain. As shown in Figure 16, optimizing first for the rigid motion prior to the full pose estimation leads to improved robustness of the tracking.

Parameters. For all our results we fix our parameters to $\omega_1 = \omega_2 = \omega_5 = 1$, $\omega_4 = 10^3$, $\omega_3 = \omega_6 = \omega_7 = 10^8$, $\omega_8 = \omega_9 = 3$. We determined these weights empirically by re-tracking multiple sequences with different sets of parameters. Our system was tested on an Intel Core i7 4GHz with NVIDIA GTX980 GPU running Ubuntu 12.02 . To run on a 60Hz RGBD device such as the PrimeSense Carmine 1.09 or the Creative Gesture Camera, we perform 1 rigid iteration and 7 full iterations, at 1.5ms per iteration. We perform closed form closest point correspondences and Jacobian computation for the fitting energies on the GPU. The number of iterations can be easily adapted to run on the new Intel RealSense 3D Camera (F200) at 120Hz or at even higher frame rates on future devices.

6. Evaluation

We refer to the video to best appreciate the realtime tracking performance of our method. Here we analyze its performance by providing a comparison to several state-of-the art solutions.

Dexter-1 Dataset [SRS*14]. Figure 17 shows a quantitative comparison with several existing methods on a publicly available data set acquired at 25 Hz. As the graph illustrates, our solution clearly outperforms the method of [TCTK14] that uses regression forest classifiers in an appearance-based approach to estimate hand poses. We also significantly improve upon

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

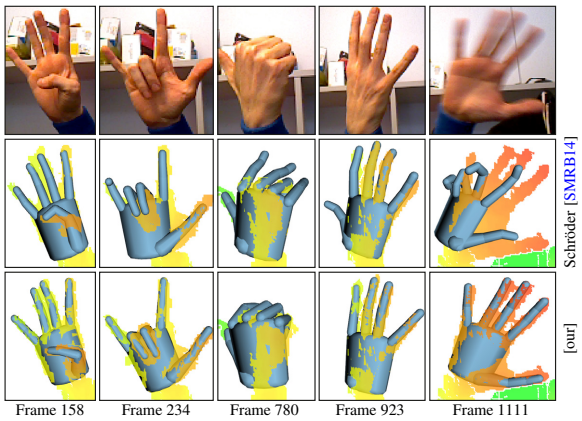


Figure 18: *A few comparison frames illustrating the difference in performance of our method compared to [SMRB14] (results provided by the authors of that paper). From left to right we can observe problems related to: correspondences to the back of the model, lack of silhouette energy (3 times) and loss of tracking due to fast motion.*

the gradient-based optimization methods of [SOT13,SRS*14] that, in addition to the depth information, use RGB data from five additional video cameras. As the dataset is acquired at 25 Hz, the performance of our algorithm (red) is suboptimal. In particular, in a single frame fingers are occasionally displaced by 2 to 3 times their radii, thus corrupting ICP correspondences. By re-initializing with finger detection as in [QSW*14] our performance considerably improves, as shown in the figure.

Subspace ICP [SMRB14]. Figure 18 shows a comparison to the model-based approach of [SMRB14]. The recorded sequences were directly processed by the authors and employed to pose our cylinder model for ease of comparison. As the figure illustrates, our method clearly outperforms this previous work. A key difference is that they optimize directly in a PCA subspace, which tends to over-constrain the solution, while we introduce a PCA data term as a regularizer, which preserves the full expressiveness of the tracking model. In addition, we introduce collision handling, apply robust norms for automatic outlier detection, and employ a more advanced correspondence search that handles self-occlusions. In combination, these factors lead to substantial improvements in tracking accuracy and robustness without compromising computational efficiency.

Convex body solver [MKO13]. We compare to this algorithm by employing the precompiled binaries from the Intel Perceptual Computing SDK. We modifed the demo application to save the recorded depth/color frames to disk while tracking. We then re-tracked this data from scratch using our technique. As illustrated in the video, as well as Figure 19,

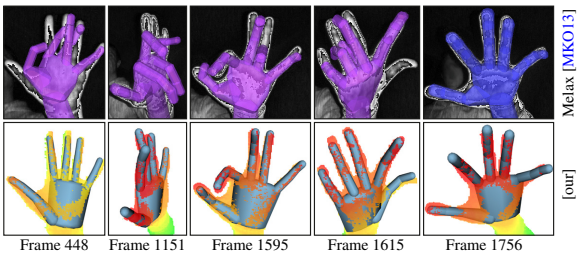


Figure 19: *Comparison to the method of [MKO13]. The full sequence can be seen in the accompanying video. We highlight a few frames that are not resolved correctly by this method, but that can be handled successfully with our solution. The last frame shows the better geometric approximation quality of the convex body model used in [MKO13] compared to our simpler cylinder model.*

our method offers a substantial increase in tracking robustness compared to [MKO13]. This can be attributed to any of the improvements we presented, but it is difficult to quantitatively identify the causes, because no control on tracking parameters nor source code is given. Their approach computes closest correspondences to the entire model, therefore not explicitly handling occlusion. The authors also proposed a technique to ensure that the model is fully contained in the 3D convex hull of the data. Note that in camera space, this amounts to constraints similar to the ones enforced by our 2D registration (Equation 3), except that the distance transform would be computed from the 2D convex hull of the silhouette image. Figure 19 (Frame 448) illustrates how our 2D registration better constrains feasible solutions. While in [MKO13] correlation between fingers is manually introduced as a *grasping bias*, our optimization is data driven and encodes correlation in a more principled way. As illustrated in Figure 19 and the video, this approach often loses tracking during complex motion. However, it is sometimes capable of recovering by sampling and then evaluating a reduced set of

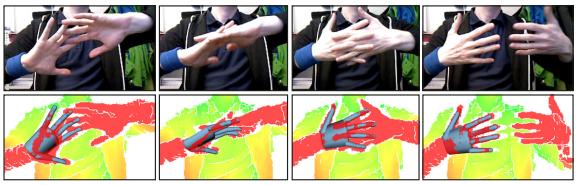


Figure 20: *Developing robust model-based tracking is essential to enable tracking of hands interacting with each other or with other objects in the environment. Here we illustrate that for our method tracking accuracy is not significantly affected even though we are not modeling the second hand. Note that such motion cannot be tracked successfully by appearance-based methods such as [TSLP14].*

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

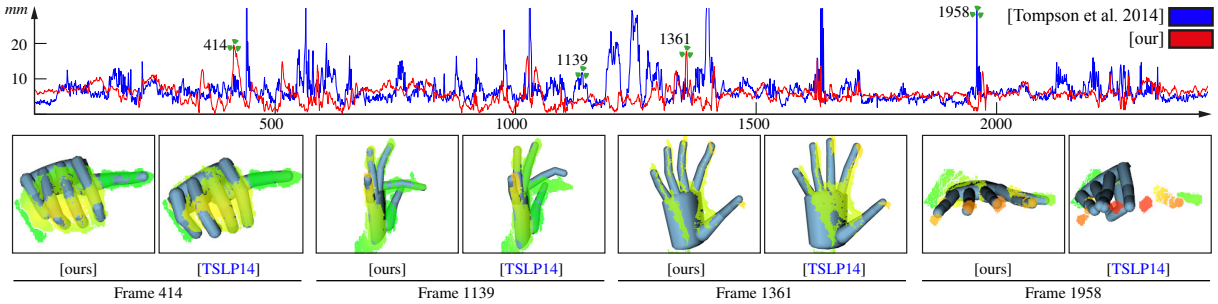


Figure 21: *Quantitative comparison to [TSLP14]. The graph shows the average root mean square tracking error w.r.t. ground truth across 2440 frames. Some frames where the accuracy of the two methods differs significantly are highlighted in the bottom row.*

poses, with an approach that is similar in spirit to [OKA11a]. One advantage of their method is the higher geometric fidelity of their convex bodies hand model compared to our cylinder model. Furthermore, our evaluation demonstrated how their more precise representation of the hand's Thenar eminence, as well as the thumb articulation, can result in more natural fitting in these regions.

Convolutional Networks [TSLP14]. Figure 21 shows a quantitative comparison with the appearance-based method of [TSLP14] on a dataset provided by the authors of that paper. Overall, the tracking quality is comparable, with a somewhat lower average error for our method. However, our solution avoids many of the high-error peaks of [TSLP14] where tracking is lost completely. An additional advantage of our approach in comparison to any of the existing appearance-based methods is that we can handle more complex interactions of two hands, since such configurations are not part of the training data sets of existing methods; see Figure 20.

Limitations. Single-camera depth acquisition yields incomplete data and as such the pose reconstruction problem is inherently ill-posed. Tracking errors can occur in certain situations as explained above when insufficient data is acquired due to occlusions or fast motion. Similarly, the resolution of the sensor limits tracking accuracy. As shown in Figure 22, when geometric features become indiscriminate, our registration approach fails. Integrating color and shading information could potentially address this issue [dLGFP11]. While our current system requires the user to wear a wristband for detection and stabilization, this could be replaced by automatic hand labeling, e.g. using forest classifiers as in [TSLP14].

Our cylinder model proved adequate for the data quality of current commodity sensors, but is overall limited in geometric accuracy, and hence might not scale with increasing sensor resolution. Also, in our current implementation the model needs to be manually adapted to the user through simple scaling operations. Without such adaptation, tracking accuracy degrades as shown in Figure 23. This user-specific adaption could be automated [TSR*14] and potentially even performed simultaneously with the realtime tracking as recently proposed for face tracking [BWP13].

The PCA model used in the prior energy is an efficient, but rather simplistic representation of the pose space. We currently do not consider the temporal order in which the hand

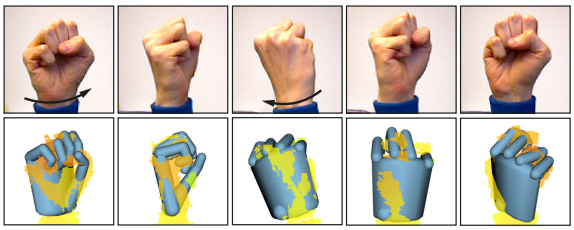


Figure 22: *Our algorithm relies on the presence of salient geometric features in the depth map. Challenging sequences like a rotating fist lack such features when acquired with current commodity depth sensors, which can result in loss of tracking.*

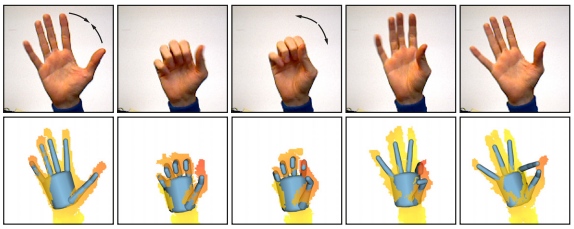


Figure 23: *When tracking with an uncalibrated model, tracking correspondences can map to data belonging to erroneous portions of the model. In the figure, the index finger remains attached to samples associated with the thumb.*

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

poses of the database have been acquired, which could potentially be exploited for more sophisticated temporal priors.

7. Conclusions

We have introduced a new model-based approach to realtime hand tracking using a single low-cost depth camera. This simple acquisition setup maximizes ease of deployment, but poses significant challenges for robust tracking. Our analysis revealed that a major source of error when tracking articulated hands are erroneous correspondences between the hand model and the acquired data, mainly caused by outliers, holes, or data popping in and out during acquisition. We demonstrate that these problems can be resolved by our new formulation of correspondence search. In combination with suitable 2D/3D registration energies and data-driven priors, this leads to a robust and efficient hand tracking algorithm that outperforms existing model- and appearance-based solutions.

In our experiments we show that our system runs seamlessly for sensors capturing data at 60 Hz. However, we can even support higher frame rates of up to 120 fps in anticipation of future sensors that have recently been announced. By fully disclosing our source code and data we ensure that our method and results are reproducible, as well as facilitating future research and product development.

We are investigating a technique for efficient automatic personalization of the tracking model to the acquired user, in order to facilitate a more seamless usage of our system across different subjects. Other examples of future efforts are robust two-hand tracking with object interactions, combinations of hand tracking with full body tracking, and integrating our hand tracking solution to new interfaces and realtime applications.

Acknowledgments

The authors thank S. Sridhar, S. Melax and J. Tompson for insightful discussions. This research is supported by the Swiss National Science Foundation grant 200021-153567. Parts of this work were supported by the Cluster of Excellence "Cognitive Interaction Technology" (EXC 277) funded by the German Research Foundation (DFG).

References

[BDS*12] BOUAZIZ S., DEUSS M., SCHWARTZBURG Y., WEISE T., PAULY M.: Shape-up: Shaping discrete geometry with projections. *Computer Graphics Forum (Proc. of the Symposium on Geometry Processing)* (2012). 14

[BTG*12] BALLAN L., TANEJA A., GALL J., VAN GOOL L., POLLEFEYS M.: Motion capture of hands in action using discriminative salient points. In *Proc. of the European Conference on Computer Vision* (2012), IEEE. 3

[BTP13] BOUAZIZ S., TAGLIASACCHI A., PAULY M.: Sparse iterative closest point. *Computer Graphics Forum (Proc. of the Symposium on Geometry Processing)* (2013). 5, 14

[BTP14] BOUAZIZ S., TAGLIASACCHI A., PAULY M.: Dynamic 2D/3D registration. *Eurographics Tutorial* (2014). 13

[Bus04] BUSS S. R.: Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation* (2004). 13

[BWP13] BOUAZIZ S., WANG Y., PAULY M.: Online modeling for realtime facial animation. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2013). 11

[CD95] CHAN T. F., DUBEY R. V.: A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators. *IEEE Transactions on Robotics and Automation* (1995). 8

[CHZ14] CAO C., HOU Q., ZHOU K.: Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2014). 1

[dLGFP11] DE LA GORCE M., FLEET D. J., PARAGIOS N.: Model-based 3D hand pose estimation from monocular video. *Pattern Analysis and Machine Intelligence* (2011). 11

[EBN*07] EROL A., BEBIS G., NICOLESCU M., BOYLE R. D., TWOMBLY X.: Vision Based Hand Pose Estimation: A Review. *Computer Vision Image Understanding* (2007). 2

[FH12] FELZENSZWALB P. F., HUTTENLOCHER D. P.: Distance transforms of sampled functions. *Theory of Computing* (2012). 6

[GPKT12] GANAPATHI V., PLAGEMANN C., KOLLER D., THRUN S.: Real-time human pose tracking from range data. In *Proc. of the European Conference on Computer Vision* (2012), Springer, pp. 738–751. 6

[HRMO12] HOYET L., RYALL K., MCDONNELL R., O'SULLIVAN C.: Sleight of hand: Perception of finger motion from reduced marker sets. In *Proc. of the Symposium on Interactive 3D Graphics and Games* (2012). 3

[KKKA12] KESKIN C., KIRAÇ F., KARA Y. E., AKARUN L.: Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *Proc. of the European Conference on Computer Vision* (2012), IEEE. 3

[KVK*14] KRUPKA E., VINNIKOV A., KLEIN B., HILLEL A. B., FREEDMAN D., STACHNIAK S.: Discriminative ferns ensemble for hand pose recognition. In *Computer Vision and Pattern Recognition* (2014), IEEE. 3

[MKO13] MELAX S., KESELMAN L., ORSTEN S.: Dynamics based 3D skeletal hand tracking. *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2013). 1, 3, 4, 6, 10

[OKA11a] OIKONOMIDIS I., KYRIAZIS N., ARGYROS A. A.: Efficient model-based 3D tracking of hand articulation using kinect. In *British Machine Vision Conference* (2011), BMVA. 1, 3, 4, 11

[OKA11b] OIKONOMIDIS I., KYRIAZIS N., ARGYROS A. A.: Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *International Conference on Computer Vision* (2011), IEEE. 3

[OKA12] OIKONOMIDIS I., KYRIAZIS N., ARGYROS A. A.: Tracking the articulated motion of two strongly interacting hands. In *Computer Vision and Pattern Recognition* (2012), IEEE. 3, 4

[OLA14] OIKONOMIDIS I., LOURAKIS M. I., ARGYROS A. A.: Evolutionary quasi-random search for hand articulations tracking. In *Computer Vision and Pattern Recognition* (2014), IEEE. 3

[QSW*14] QIAN C., SUN X., WEI Y., TANG X., SUN J.: Real-time and robust hand tracking from depth. In *Computer Vision and Pattern Recognition* (2014), IEEE. 2, 3, 4, 5, 6, 9, 10

[RKEK13] ROMERO J., KJELLSTRÃŰM H., EK C. H., KRAGIC

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

D.: Non-parametric hand pose estimation with object context. *Proc. Image and Vision Computing* (2013). 3

[SFC*11] SHOTTON J., FITZGIBBON A., COOK M., SHARP T., FINOCCHIO M., MOORE R., KIPMAN A., BLAKE A.: Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition* (2011), IEEE. 1, 4

[SMRB14] SCHRÖDER M., MAYCOCK J., RITTER H., BOTSCH M.: Real-time hand tracking using synergistic inverse kinematics. In *International Conference on Robotics and Automation* (2014), IEEE. 1, 3, 4, 7, 8, 10

[SOT13] SRIDHAR S., OULASVIRTA A., THEOBALT C.: Interactive markerless articulated hand motion tracking using RGB and depth data. In *International Conference on Computer Vision* (2013), IEEE. 3, 9, 10

[SRS*14] SRIDHAR S., RHODIN H., SEIDEL H.-P., OULASVIRTA A., THEOBALT C.: Real-time hand tracking using a sum of anisotropic gaussians model. In *International Conference on 3D Vision (3DV)* (2014), IEEE. 1, 3, 9, 10

[TCTK14] TANG D., CHANG H. J., TEJANI A., KIM T.-K.: Latent regression forest: Structured estimation of 3D articulated hand posture. In *Computer Vision and Pattern Recognition* (2014), IEEE. 3, 9

[TSLP14] TOMPSON J., STEIN M., LECUN Y., PERLIN K.: Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.* (2014). 1, 2, 3, 4, 9, 10, 11

[TSR*14] TAYLOR J., STEBBING R., RAMAKRISHNA V., KESKIN C., SHOTTON J., IZADI S., HERTZMANN A., FITZGIBBON A.: User-specific hand modeling from monocular depth sequences. In *Computer Vision and Pattern Recognition* (2014). 11

[TYK13] TANG D., YU T.-H., KIM T.-K.: Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *International Conference on Computer Vision* (2013), IEEE. 3

[WMZ*13] WANG Y., MIN J., ZHANG J., LIU Y., XU F., DAI Q., CHAI J.: Video-based hand manipulation capture through composite motion control. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2013). 3

[WP09] WANG R. Y., POPOVIC J.: Real time hand tracking with a colored glove. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2009). 3

[WPP11] WANG R. Y., PARIS S., POPOVIC J.: 6D hands: Markerless hand tracking for computer aided design. *Proc. of the Symposium on User Interface Software and Technology* (2011). 3

[WZC12] WEI X., ZHANG P., CHAI J.: Accurate realtime full-body motion capture using a single depth camera. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* (2012). 1, 2, 3, 5, 6, 8, 9

[YZW*13] YE M., ZHANG Q., WANG L., ZHU J., YANG R., GALL J.: A survey on human motion analysis from depth data. *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications* (2013). 2

[ZCX12] ZHAO W., CHAI J., XU Y.-Q.: Combining marker based mocap and RGBD camera for acquiring high-fidelity hand motion data. In *Proc. of the EG/SIGGRAPH Symposium on Computer Animation* (2012). 3

[ZSZ*14] ZHANG P., SIU K., ZHANG J., LIU C. K., CHAI J.: Leveraging depth cameras and wearable pressure sensors for full-body kinematics and dynamics capture. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* (2014). 2

Appendix A: Projective v.s. subspace PCA

In Equation 6, minimizing E_{pose} over $\tilde{\boldsymbol{\theta}}$ has a closed form solution:

$$\tilde{\boldsymbol{\theta}} = (\omega_5 \boldsymbol{\Sigma}^2 + \omega_4 \mathbf{I})^{-1} (\omega_4 \Pi_{\mathcal{P}}^T (\boldsymbol{\theta} - \boldsymbol{\mu})).$$

We can therefore rewrite our data-driven energy only as a function of $\boldsymbol{\theta}$ as

$$E_{\text{pose}} = \omega_4 \| (\boldsymbol{\theta} - \boldsymbol{\mu}) - \Pi_{\mathcal{P}} \mathbf{M} \Pi_{\mathcal{P}}^T (\boldsymbol{\theta} - \boldsymbol{\mu}) \|_2^2,$$

where $\mathbf{M} = \omega_4 (\omega_5 \boldsymbol{\Sigma}^2 + \omega_4 \mathbf{I})^{-1}$. Our formulation does not only allow the solution to stay close to the pose space, but also penalizes unlikely poses replacing the conventional orthogonal projection matrix $\Pi_{\mathcal{P}} \Pi_{\mathcal{P}}^T$ by a matrix $\Pi_{\mathcal{P}} \mathbf{M} \Pi_{\mathcal{P}}^T$ taking into account the PCA standard deviation. Note that when $\omega_5 = 0$ we retrieve the orthogonal projection $\Pi_{\mathcal{P}} \Pi_{\mathcal{P}}^T$.

Appendix B: Jacobians

Perspective projection Jacobian. The Jacobian of the perspective projection is a $[2 \times 3]$ matrix depending from the focal length of the camera $\mathbf{f} = [f_x, f_y]$ and the 3D position $\mathbf{x}$ at which it is evaluated [BTP14]:

$$\mathbf{J}_{\text{persp}}(\mathbf{x}) = \begin{bmatrix} f_x/\mathbf{x}_z & 0 & -\mathbf{x}_x f_x/\mathbf{x}_z^2 \\ 0 & f_y/\mathbf{x}_z & -\mathbf{x}_y f_y/\mathbf{x}_z^2 \end{bmatrix}$$

Skeleton Jacobian. The skeleton Jacobian $\mathbf{J}_{\text{skel}}(\mathbf{x})$ is a $[3 \times 26]$ matrix. For each constraint, the bone identifier $b = id(\mathbf{x})$ associated to each 3D point $\mathbf{x}$ determines the affected portion of the kinematic chain. That is, it identifies the non-zero columns of $\mathbf{J}_{\text{skel}}(\mathbf{x})$. As discussed in [Bus04], the i-th column of $\mathbf{J}_{\text{skel}}(\mathbf{x})$ contains the linearization of i-th joint about $\mathbf{x}$.

Appendix C: Approximation using linearized function.

To approximate the following energies, we approximate $E = \|\mathbf{f}(\mathbf{x})\|_2^2$ by linearizing $\mathbf{f}(\mathbf{x})$ as

$$\mathbf{f}(\mathbf{x} + \delta\mathbf{x})|_{\mathbf{x}} \approx \mathbf{f}(\mathbf{x}) + \mathbf{J}(\mathbf{x}) \delta\mathbf{x}.$$

The approximation is then expressed as

$$\bar{E} = \|\mathbf{f}(\mathbf{x}) + \mathbf{J}(\mathbf{x}) \delta\mathbf{x}\|_2^2. \tag{10}$$

Joint bounds. The joint bounds energy can be written as

$$\bar{E}_{\text{bound}} = \omega_7 \sum_{\theta_i \in \boldsymbol{\theta}} \overline{\chi}(i) (\delta\boldsymbol{\theta}_i + \boldsymbol{\theta}_i - \overline{\boldsymbol{\theta}}_i)^2 + \underline{\chi}(i) (\delta\boldsymbol{\theta}_i + \boldsymbol{\theta}_i - \underline{\boldsymbol{\theta}}_i)^2$$

Temporal coherence. To compute the velocity $\dot{\mathbf{k}}(\boldsymbol{\theta})$ and the acceleration $\ddot{\mathbf{k}}(\boldsymbol{\theta})$ of a point $\mathbf{k}$ attached to the kinematic chain, we use finite differences. The linearization of the temporal energy becomes

$$\bar{E}_{\text{temporal}} = \omega_8 \sum_{\mathbf{k} \in \mathcal{K}} \|\mathbf{J}_{\text{skel}}(\mathbf{k}) \delta\boldsymbol{\theta} + (\mathbf{k} - \mathbf{k}_{t-1})\|_2^2 + \omega_9 \sum_{\mathbf{k} \in \mathcal{K}} \|\mathbf{J}_{\text{skel}}(\mathbf{k}) \delta\boldsymbol{\theta} + (\mathbf{k} - 2\mathbf{k}_{t-1} + \mathbf{k}_{t-2})\|_2^2,$$

where $\mathbf{k}_{t-1}$ and $\mathbf{k}_{t-2}$ are the position of such points from the two previously optimized frames.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

Data-driven (PCA). The data-driven projection energy can be rewritten as

$$\bar{E}_{\text{pose}} = \omega_4 \left\| (\mathbf{I} - \Pi_{\mathcal{P}} \mathbf{M} \Pi_{\mathcal{P}}^T)(\delta\boldsymbol{\theta} + \boldsymbol{\theta} - \boldsymbol{\mu}) \right\|_2^2.$$

Appendix D: Approximation using Linearized ℓ_2 Distance.

To approximate the following energies, we first reformulate the quadratic form $E = \|\mathbf{f}(\mathbf{x})\|_2^2$ as $E = (\|\mathbf{f}(\mathbf{x})\|_2)^2$. We then linearize the ℓ_2 norm $\|\mathbf{f}(\mathbf{x})\|_2$ as

$$\|\mathbf{f}(\mathbf{x} + \delta\mathbf{x})\|_2|_{\mathbf{x}} \approx \|\mathbf{f}(\mathbf{x})\|_2 + \frac{\mathbf{f}(\mathbf{x})^T}{\|\mathbf{f}(\mathbf{x})\|_2} \mathbf{J}(\mathbf{x})\delta\mathbf{x}.$$

The approximation is then expressed as

$$\bar{E} = \left(\|\mathbf{f}(\mathbf{x})\|_2 + \frac{\mathbf{f}(\mathbf{x})^T}{\|\mathbf{f}(\mathbf{x})\|_2} \mathbf{J}(\mathbf{x})\delta\mathbf{x} \right)^2.$$

When the energy is of the form $E = \|\mathbf{x} - \Pi(\mathbf{x})\|_2^2$ where $\Pi(\mathbf{x})$ is a projection operator, Bouaziz et al. [BDS*12] showed that $\mathbf{f}(\mathbf{x})^T \mathbf{J}(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T$. In this case, the approximate energy can be simplified as

$$\bar{E} = \left(\|\mathbf{f}(\mathbf{x})\|_2 + \frac{\mathbf{f}(\mathbf{x})^T}{\|\mathbf{f}(\mathbf{x})\|_2} \delta\mathbf{x} \right)^2.$$

Contrary to the approximation in Equation 10, the Jacobian of the projection function does not need to be known. This formulation is useful as the approximation in the equation above only needs to evaluate the projection function and therefore allows to use arbitrarily complex projection functions.

Point cloud alignment. We linearize the point cloud alignment energy as

$$\bar{E}_{\text{3D}} = \omega_1 \sum_{\mathbf{x} \in \mathcal{X}_s} \omega_{\text{re}} (\mathbf{n}^T (\mathbf{J}_{\text{skel}}(\mathbf{y})\delta\boldsymbol{\theta} + d))^2,$$

where $\mathbf{y} = \Pi_{\mathcal{M}}(\mathbf{x}, \boldsymbol{\theta})$ is the closest point from $\mathbf{x}$ on the hand model $\mathcal{M}$ with hand pose $\boldsymbol{\theta}$. $\mathbf{n}$ is the surface normal at $\mathbf{y}$, and $\mathbf{d} = (\mathbf{y} - \mathbf{x})$. As we minimize the ℓ_2 norm we use a weight $\omega_{\text{re}} = 1/\|\mathbf{d}\|_2$ in an iteratively re-weighted least squares fashion [BTP13].

Silhouette alignment. The silhouette energy is expressed in screen space, and therefore employs the perspective projection Jacobian $\mathbf{J}_{\text{persp}}(\mathbf{x})$, where $\mathbf{x}$ is the 3D location of a rendered silhouette point $\mathbf{p}$. Similarly to the point cloud alignment the linearization can be expressed as

$$\bar{E}_{\text{2D}} = \omega_2 \sum_{\mathbf{p} \in \mathcal{S}_r} (\mathbf{n}^T (\mathbf{J}_{\text{persp}}(\mathbf{x}) \mathbf{J}_{\text{skel}}(\mathbf{x})\delta\boldsymbol{\theta} + d))^2,$$

where $\mathbf{d} = (\mathbf{p} - \mathbf{q})$ with $\mathbf{q} = \Pi_{\mathcal{S}_s}(\mathbf{p}, \boldsymbol{\theta})$, and $\mathbf{n}$ is the 2D normal at the sensor silhouette location $\mathbf{q}$.

Collision. Figure 24 illustrates the necessary notation with a 2D example, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the end-points of the shortest segment between the two cylinders axes. The linearized energy is defined as

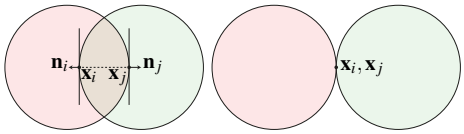


Figure 24: *(left) Collision constraints definition, deepest penetration points marked as $\mathbf{x}_i, \mathbf{x}_j$. (right) When the collision energy is minimized in isolation the penetration points are co-located.*

$$\bar{E}_{\text{coll.}} = \omega_6 \sum_{\{i,j\}} \chi(i,j) \left(\mathbf{n}_i^T \left((\mathbf{J}_{\text{skel}}(\mathbf{x}_i) - \mathbf{J}_{\text{skel}}(\mathbf{x}_j))\delta\boldsymbol{\theta} + d \right) \right)^2$$

where $\mathbf{n}_i$ is the surface normal at $\mathbf{x}_i$ (as shown in Figure 24), and $\mathbf{d} = (\mathbf{x}_i - \mathbf{x}_j)$.

Appendix E: Non-Linear Least Squares Optimization

To solve our optimization problem we use a Levenberg-Marquardt approach. We iteratively solve Equation 1 using the approximate energies described in Appendix B through Appendix D leading to a damped least squares minimization

$$\min_{\delta\boldsymbol{\theta}} \bar{E}_{\text{3D}} + \bar{E}_{\text{2D}} + \bar{E}_{\text{wrist}} + \bar{E}_{\text{pose}} + \bar{E}_{\text{kin.}} + \bar{E}_{\text{temp.}} + \bar{E}_{\text{damp}},$$

and update our hand pose using the update $\boldsymbol{\theta} = \boldsymbol{\theta} + \delta\boldsymbol{\theta}$. Note that since our energies are written in the form:

$$\Sigma_i \bar{E}_i = \Sigma_i \|\mathbf{J}_i \delta\boldsymbol{\theta} - \mathbf{e}_i\|_2^2,$$

our solve can be re-written as

$$\delta\boldsymbol{\theta} = \left(\Sigma_i \mathbf{J}_i^T \mathbf{J}_i \right)^{-1} \left(\Sigma_i \mathbf{J}_i^T \mathbf{e}_i \right) = 0. \tag{11}$$

To stabilize the optimization, we introduce a damping energy $\bar{E}_{\text{damp}} = \lambda \|\delta\boldsymbol{\theta}\|_2^2$, where $\lambda = 100$.

Appendix F: CPU/GPU Optimization

Our technique elegantly de-couples the components of our optimization on CPU and GPU. With regards to Figure 3 only large-scale and trivially parallelizable tasks, like the computation of constraints associated with 2D/3D ICP correspondences are performed on GPU, while all others run efficiently on a *single* CPU thread. In particular, the inversion in Equation 11 is performed on CPU by Cholesky factorization (Eigen3). As the final solve is performed on CPU, we designed our optimization to minimize memory transfers between CPU/GPU. First of all, note that although at each iteration we need to render an image of the cylinder model, the texture is already located on the GPU buffers. Furthermore, although the large ($\approx 20k \times 26$) Jacobian matrices associated with E_{3D} and E_{2D} are assembled on the GPU, a CuBLAS kernel is used to compute the much smaller (26×26, 26×1) matrices $\mathbf{J}_i^T \mathbf{J}_i$ and $\mathbf{J}_i^T \mathbf{e}_i$. Only these need to be transferred back to CPU for each solver iteration.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.