Data-Driven Animation of Hand-Object Interactions

Henning Hamer[1] Juergen Gall[1] Raquel Urtasun[2] Luc Van Gool[1,3]

[1]Computer Vision Laboratory
ETH Zurich

[2]TTI Chicago

[3]ESAT-PSI / IBBT
KU Leuven

{hhamer,gall,vangool}@vision.ee.ethz.ch rurtasun@ttic.edu luc.vangool@esat.kuleuven.be

***Abstract*— Animating hand-object interactions is a frequent task in applications such as the production of 3d movies. Unfortunately this task is difficult due to the hand's many degrees of freedom and the constraints on the hand motion imposed by the geometry of the object. However, the causality between the object state and the hand's pose can be exploited in order to simplify the animation process. In this paper, we present a method that takes an animation of an object as input and automatically generates the corresponding hand motion. This approach is based on the simple observation that objects are easier to animate than hands, since they usually have fewer degrees of freedom. The method is data-driven; sequences of hands manipulating an object are captured semi-automatically with a structured-light setup. The training data is then combined with a new animation of the object in order to generate a plausible animation featuring the hand-object interaction.**

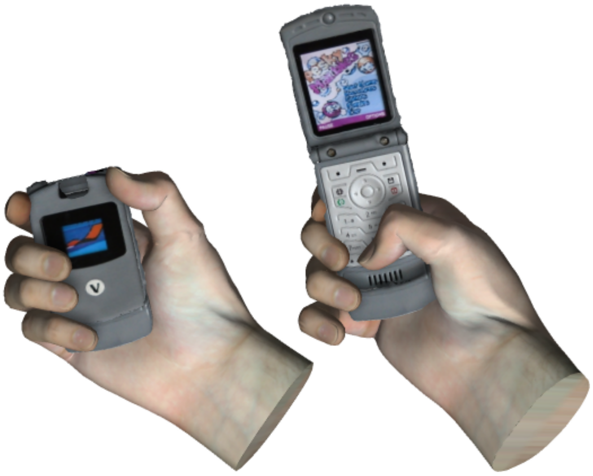


Fig. 1. Two frames of an animation demonstrating the usage of a clamshell phone. The hand animation is automatically generated from the given phone animation.

I. Introduction

When humans interact with objects, hand and object motions are strongly correlated. Moreover, a hand manipulates an object usually with a purpose, changing the state of the object. Vice versa an object has certain affordances [6], i.e., it suggests a certain functionality. Consider the clamshell phone in Fig. 1 as an introductory example. Physical forces are applied to pick up such a phone and to open it. Once the phone is opened, the keys with the digits suggest dialing a number.

The affordances of an object have the potential to ease hand animation in the context of hand-object interaction, e.g., given the clamshell phone and a number to dial, the necessary hand motions to make a call can be synthesized. This is particularly interesting when the object has fewer degrees of freedom (DOFs) than the hand (e.g., opening the phone requires just a one-dimensional rotation) or when the DOFs are largely independent (like in the case of the separate digits of the phone). Animating such an object is easier for an artist than animating the hand or both. Ideally, simple scripting of object state changes infers a complete hand animation to carry out these changes.

Inspired by these considerations, we present a method to animate a manipulating hand conditioned on an animation of the manipulated object. The approach is data-driven, so we require that the object has previously been observed during manipulation. A training phase involves a semi-automatic acquisition of hand poses and object poses from structured-light data. The pose of an object always comprises its translation and rotation. In case of articulated objects or objects consisting of several connected rigid parts, the object's pose also includes information regarding the arrangement of its parts. Based on the captured hand and the tracked object, we infer 1) the various states of the object during manipulation, 2) the hand configurations that cause object state transitions, and 3) the spatio-temporal correlations between key hand poses and key object poses. For instance, the state of the phone can be either closed or open and a specific temporal hand movement is required for opening and closing. Data acquisition and training is required only once for a new object.

For animation, the object pose and contact points optionally created by the artist are used to generate hand poses for key frames. The hand pose transitions that have been observed during training then form the basis for hand pose interpolation to obtain a plausible hand-object animation. With this technique an artist can quickly produce a great variety of different animations without the need of acquiring new data.

Compared to previous work on hand-object animation [11], [5], [18], [13], [14], [15], [16], our approach handles articulated objects and hand-object interactions with significant changes of contact points over time, e.g., opening

The authors gratefully acknowledge support through the EC Integrated Project 3D-Coform.

a clamshell phone and dialing a specific number as shown in Fig. 1. It is neither limited to rigid objects nor to a specific musical instrument. Furthermore, the relevant object states and the corresponding hand poses are inferred from training data within a spatio-temporal context. Our data acquisition is non-invasive because we use a marker-less vision system.

II. Related Work

A. Hand-Object Interaction in Robotics and Vision

A taxonomy of hand poses with regard to the grasping of objects was provided in [3]. Grasp quality has been studied in robotics [2]. For example, given a full 3d model and a desired grasp, the stability of grasping can be evaluated based on pre-computed grasp primitives [17]. In [22], 3d grasp positions are estimated for a robotic hand from image pairs in which grasp locations are identified. For this, a 2d grasp point detector is trained on synthetic images.

In [10], manipulative hand gestures are visually recognized using a state transition diagram that encapsulates task knowledge. The person has to wear special gloves, and gestures are simulated without a real object. [4] recognizes grasps referring to the grasp taxonomy defined in [3], using a data glove. In [9], task relevant hand poses are used to build a low dimensional hand model for marker-less grasp pose recognition. In [12], visual features and the correlation between a manipulating hand and the manipulated object are exploited for both better hand pose and object *recognition*. Recently, a real-time method was presented in [20] that compares observed hand poses to a large database containing hands manipulating objects. In contrast, our method for hand pose estimation is not constrained to a set of examples and comes with the capability to generalize.

B. Animating Hand-Object Interaction

Many approaches in computer graphics are concerned with realistic hand models. For example, in [1] an anatomically based model is animated by means of muscle contraction. However, there has been less work with respect to hand-object interaction. Some approaches address the synthesis of realistic static grasps on objects [14] or grasp-related hand motion [18], [13], [15], [16]. Li et al. [14] treat grasp synthesis as a 3d shape matching problem: grasp candidates are selected from a large database by matching contact points and surface normals of hands and objects. Pollard and Zordan [18] propose a grasp controller for a physically based simulation system. To obtain realistic behavior, the parameters of the controller are estimated from motion sequences captured with markers. A similar method is used by Kry and Pai [13] where hand motion and contact forces are captured to estimate joint compliances. New interactions are synthesized by using these parameters for a physically based simulation. Recently, Liu [15], [16] formulated the synthesis of hand manipulations as an optimization problem where an initial grasping pose and the motion of the object are given. Besides grasping motions, hand motions for musical instruments have also been modeled [11], [5]. In these works, a hand plays a specific musical instrument, e.g., violin or guitar.

We now classify our approach and at the same time point out differences to the other works.

1) Our approach is data-driven as we exploit observations of real manipulations to ease the synthesis of new animations. This is a common strategy with regard to the animation of manipulating hand motion, since manual modeling of hand-object interaction does not achieve realistic results. However, in contrast to our method most data-driven systems use invasive techniques like markers or gloves [14], [18], [13].
2) We consider not only grasping but also manipulations where contact points change dramatically during hand-object interaction. Works like [11], [5] in which musical instruments are played are other notable exceptions.
3) The hand is controlled by the state of the manipulated object. In [15], [16] a hand is also controlled by means of the manipulated object, but their objects are not articulated and typically only grasped. Moreover, an initial grasp has to be defined which is not necessary with our method. In [11], [5], a hand plays violin or guitar. The hand is somehow controlled by the object (a certain musical score is requested), but in those works the object state does not involve a significant geometric deformation of the object. [18] also do not deal with articulated objects, and the hand state is determined by a grasp controller and not by a manipulated object.

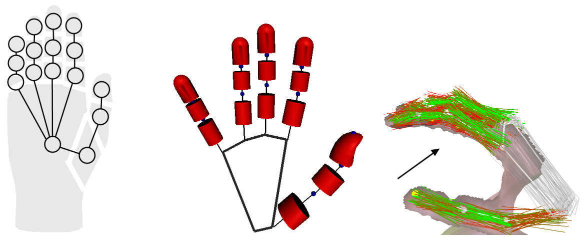
III. Learning by Human Demonstration

Our goal is to generate animations of hands manipulating an object by animating the object only. To this end, we fuse several types of information. On the one side, there is the object animation created for example in Maya. On the other side, we use information regarding the manipulation of the respective object (hand poses in relation to the object, possible articulations of the object, timing information). The latter is obtained from human demonstration.

A. Capturing Object Manipulation from Range Data

All our observations are retrieved by a structured-light setup, delivering dense 2.5d range data and color information in real-time [23]. Using this setup we observe the manipulation of a specific object by a hand and gather information regarding a) the fully articulated hand pose and b) the object's surface geometry and the object pose.

Hand Pose Our method requires knowledge about the manipulating hand. For this, we use a hand tracker [8] that operates on a graphical model in which each hand segment is a node (Fig. 2(a)). First, the observed depth information is compared to the hand model (Fig. 2(b)) to compute a data term for each hand segment. Then, anatomical constraints between neighboring hand segments are introduced via compatibility terms. In each time step, samples are drawn locally around the hand segment states of the last time step (Fig. 2(c)), the observation model is evaluated, and belief propagation



(a) Graph (b) Hand Model (c) Local samples

Fig. 2. Hand tracking. (a) Graphical model for inference. (b) Hand model with a skeleton and ruled surfaces for the skin. (c) Depth data and hand segment samples. Color encodes relative observation likelihood: green is highest, red is lowest. The palm has uniform observation likelihood. An arrow indicates the viewing direction of the camera.

is performed[1] to find a globally optimal hand pose. For initialization, the hand pose is determined manually in the first frame.

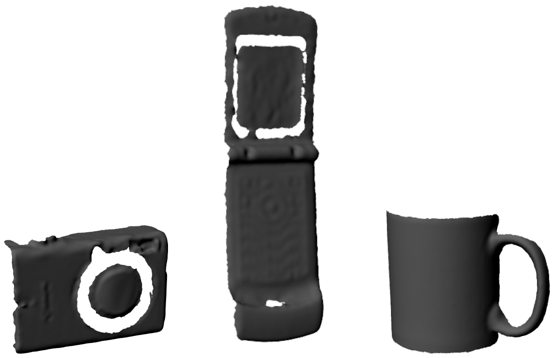
Object occlusions complicate hand tracking. Conceptually, the tracker is designed to handle this aggravated scenario. However, there are still situations in which the hand pose cannot be resolved correctly because the observation is too corrupted. Hence, we manually label the segment positions in some key frames, making the training process semi-automatic.

Object Geometry and Pose As range scans of the object are captured continuously, we register these scans online and build up a coherent mesh of the already observed parts of the surface, as demonstrated in [21]. Example meshes obtained by this procedure are shown in Fig. 3. With the partial mesh of the object available, we determine in an offline process the object's 6d pose (translation and orientation) for each frame of a sequence containing the object and some manipulation. This is done by fitting the mesh to the observation with ICP.

For articulated objects we produce a separate mesh for each extreme articulation. In the example of the phone one mesh represents the *phone closed* state and a second one the *phone open* state. We then fit the respective mesh to the data with ICP, depending on the object state. However, this leaves us without a registration during object state transitions from one state to the other.

B. Identifying Articulated Object States

There is a strong dependency between the state of an articulated object and its usage. For instance a closed clamshell phone is treated differently than an open one. Identifying the articulated states of an object manipulated in front of the structured-light setup is key to extracting manipulation knowledge. We approach the issue with a distance matrix for all frames of an observed sequence. To measure the distance between two range scans S_1 and S_2, we first remove all 3d points that have skin color. For each remaining point p of scan S_1, the closest point q_p in S_2 is found after global ICP alignment. To obtain a symmetric measure, we compute the smallest distances in both directions and take the sum as the distance:

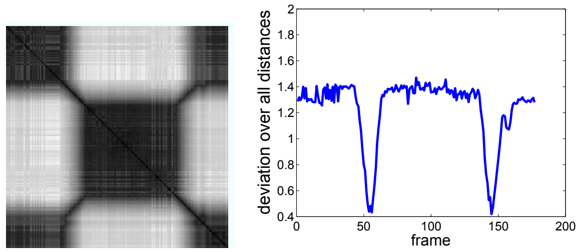


(a) Camera (b) Clamshell phone (c) Cup

Fig. 3. Partial object meshes created by integrating several range scans.

$$d(S_1, S_2) = \sum_{p \in S_1} \|p - q_p\| + \sum_{q \in S_2} \|q - p_q\|. \tag{1}$$

Fig. 4(a) shows the distance matrix for a sequence of 177 frames in which the camera is manipulated. The lens of the camera first emerges and then returns to the original position. The two different states - lens completely moved in or out - are visible. To obtain a significant measure for frame segmentation, we compute the standard deviation for each column of the distance matrix (Fig. 4(b)). High values indicate frames in which the object is in one of the two binary states.



(a) Distance matrix (b) Standard deviation

Fig. 4. Detecting object states in observed data. (a) Distance matrix for a sequence of 177 frames in which the camera is manipulated. Dark means similar. (b) Standard deviation of the columns of the distance matrix.

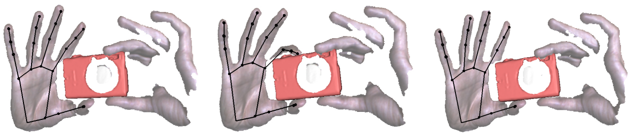
C. Transition Intervals of Object and Hand

A manipulating hand is typically most active when it causes the object to pass from one state to another (object state transition). In order to find the hand poses that produce a certain object transition, we look for corresponding hand transition intervals. In the easiest case, hand transition intervals are temporally identical with object transition intervals. This is usually the case when the object is physically forced into the new state, e.g., the clamshell phone is opened by a push. However, hand transition intervals can also differ temporally from the object transitions.

Fig. 5 shows three frames of the camera sequence analyzed in Section III-B. The tracked hand pushes an activation

[1] using libDAI v0.2.2 (http://www.libdai.org)

button on the camera, and thereby causes the first object state transition visible in Fig. 4(b). All three frames are relevant and should be reflected in the animation. The camera has a time delay, and by the time the lens state changes the finger already starts to move upwards again.



(a) frame 10 (b) frame 50 (c) frame 75

Fig. 5. Three frames showing an observed hand that pushes an activation button on the camera. The black stick-model skeleton illustrates the estimated hand pose. The registered mesh of the camera is drawn in red. In this case we excluded the lens so that the same mesh can be registered throughout the complete sequence.

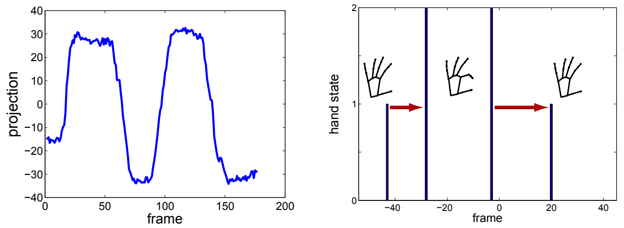
More generally speaking, hand motion performed for object manipulation can be approximated by a sequence of characteristic key poses, each with some temporal offset with respect to the object state transition. We assume that significant hand poses are those surrounding intervals of rapid change in hand state space (excluding wrist translation and rotation). To reduce noise from this high dimensional state space, we apply *principal component analysis* (PCA).

Fig. 6(a) shows the projection of the hand poses of the camera sequence to the first principal component. The two relevant hand states are visible at -30 and 30. The figure can be interpreted as follows: the index finger of the manipulating (right) hand is extended in the beginning of the sequence. It then approaches the activation button of the camera, presses the button, and then raises again. This causes the lens of the camera to emerge (zoom). This hand motion is shortly after repeated, this time with the purpose to make the lens go back. Fig. 6(b) focuses on frames 0 to 100 of the sequence and the first object state transition. The beginning and end of each transition interval of the hand are expressed relative to the middle of the object state transition, i.e., the lens is in the middle of emersion (Fig. 4(b)). Finally, the tracked sequence is divided into a series of hand transition intervals indicated by the arrows in Fig. 6(b).

IV. Animation Framework

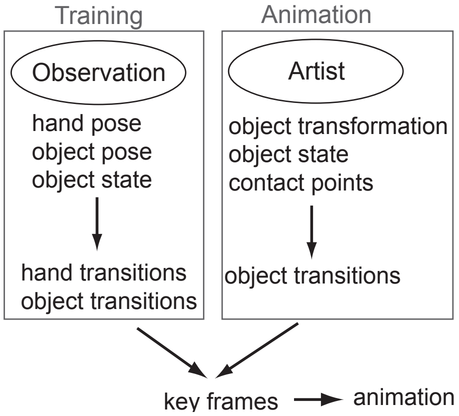
Fig. 7 gives an overview of our method. The previous section shows how to acquire and process training examples (Fig. 7 (left) - training). We now describe how to create a new animation. First, the artist chooses a hand to be animated, and *hand retargeting* is performed. Then the artist defines an object animation (Fig. 7 (right) - animation). Finally, the training information and the artist's input are combined to generate a new animation (Fig. 7 (bottom)).

A. Hand Retargeting



(a) Hand PCA (b) Hand Transitions

Fig. 6. (a) The two states of the hand are indicated by the values -30 (index finger extended) and 30 (index finger flexed). The sequence starts with the extended index finger (frame 0). Around frame 20, the finger flexes to press the activation button on the camera, causing the lens to emerge. After frame 50, the index finger begins to extend again. The same hand motion is repeated, starting near frame 90, to make the lens go back again. (b) The beginning and end of each transition interval of the hand are expressed relative to the middle of the object state transition, i.e., the lens is in the middle of emersion. Red arrows indicate the transition from extended to flexed index finger and vice versa.



Training

Observation

hand pose
object pose
object state

hand transitions
object transitions

Animation

Artist

object transformation
object state
contact points

object transitions

key frames

animation

Fig. 7. Animation procedure. Observations of a new object are processed only once for training. A new object animation can be created in Maya, optionally including contact points. The training data is then used together with the object's animation to generate an animation featuring hand-object interaction.

All hand poses estimated from the structured-light data exhibit the anatomical dimensions of the demonstrating hand, and are specified using the tracking hand model which consists of local hand segments. For visualization we use a more accurate hand model composed of a 3d scan of a hand controlled by a 26 DOF forward-kinematics skeleton in Maya (see Fig. 1).

To retarget the observed hand poses to the new anatomy, we adapt the length of the phalanges and the proportions of the palm. In particular, we preserve the position of the finger tips in space and elongate or shorten the finger segments from farthest to closest to the palm, respecting joint angles. After this, the proportions of the palm, i.e., the relative positions of the attachment points of the five fingers, are set. Finger and palm adaptation may create gaps between the fingers and the palm. We therefore apply the rigid motion to the palm that minimizes these gaps. After adapting the anatomy, we map the hand poses from the state space of the tracking hand model to that of the Maya skeleton.

B. Object Animation

Based on partial meshes created by integrating several range scans (Fig. 3), we created three Maya models (Fig. 8). In the case of the phone, a joint was added to enable

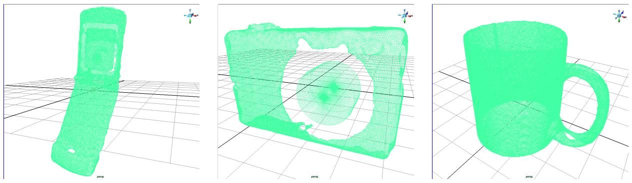


Fig. 8. Rough object models created in Maya on the basis of the partial meshes. The phone contains a joint controlling the angle between main body and display. For the camera a cylinder was added to represent the lens. The mesh of the cup was created by mirroring and is almost closed.

the animation of the opening and closing process. For the camera, a polygonal cylinder represents the lens. As input to our system, the artist creates an animation of the object, varying translation, rotation, and the object's articulation over time. Articulation is represented by continuous parameters, e.g., the translation of the lens of the camera or the angle of the joint of the phone. In addition, the artist can optionally specify contact points between the hand and the model in desired key frames, e.g., when the animated hand should dial a specific digit.

C. Combining all Information

At this point, the information from the training data and the artist can be combined. Contact points defined by the artist are used to compute hand key poses. These key poses are derived taking into consideration 1) the desired contact points and 2) all hand poses observed for a certain articulated pose of the object. Fig. 9 shows all hand poses of a training sequence observed while the clamshell phone is open.

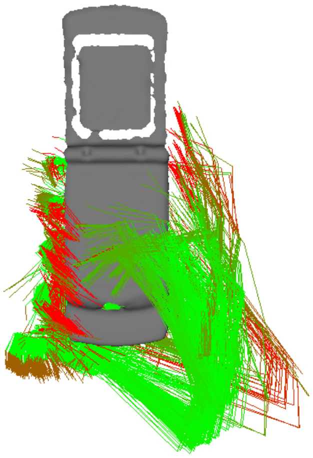


Fig. 9. Hand poses observed in a training sequence while the phone is open. Red samples have a lower probability and are penalized during optimization.

We seek the hand pose which is close to the observed hand poses and realizes the contact best without intersecting the object's geometry. We perform inference by running belief propagation on the hand graph. Note that this inference procedure is the same used for tracking, however, the local likelihood enforces the criteria mentioned above and not conformity with depth data. See [7] for details.

Other key frames result from the defined object state transitions (Section IV-B). Their midpoints determine the timing of the corresponding hand pose transitions observed in Section III-C. Hand pose interpolation between key frames of the hand is performed as follows:

- If the animator wants to pause in a certain object state this leads to a simple freeze.
- Between key frames specified via contact points, a linear interpolation regarding the joint angles of the animated hand is applied. The time warping is non-linear and reflects the assumption that human hands at first approach their targets fast but slow down in the end [19]. We transfer this observation to individual hand segments. The duration of the transition is normalized to $t = [0, 1]$. The angle vector ϕ contains three angles with respect to the rotation of a certain joint and is defined by
$$\phi_t = \phi_{t=0} + \sqrt{t} \cdot (\phi_{t=1} - \phi_{t=0}). \quad (2)$$
The square root of t causes a decrease of the speed as t approaches 1.
- For hand transitions between key frames caused by object state transitions, we follow a two-stage procedure. Initially, we temporally scale the observed hand transition, to synchronize it with the artist's prescription. However, this is more or less a replay and does not suffice. Observed transitions are characterized by a key frame at their start and end. An ending key frame and the subsequent starting key frame may be quite different, hence, the final hand animation has to blend out such abrupt changes. We formulate this as an optimization problem that strikes a balance between staying close to the observed transitions, while producing good blends between their boundaries:
$$\operatorname*{argmin}_{d\Theta_t} \sum_t \|d\Theta_t - d\tilde{\Theta}_t\|^2 + \alpha \cdot \|\Theta_0 + \sum_t d\Theta_t - \Theta_1\|^2.$$
A transition is split into increments $d\Theta_t$, and $d\tilde{\Theta}_t$ represents the corresponding increments of the stretched replay. Hence, the first term enforces compliance with the replay. The second term ensures the blending. Θ_0 and Θ_1 are the joint angles at the start of two subsequent transitions. α is a user parameter and controls the trade-off between compliance with the stretched replay and smooth blending. In our experiments we set α to 10.

V. Results

We now present results of the proposed method with respect to the three objects introduced earlier: the camera, the cup, and the phone. We also discuss the additional example of a mortar and the appendant pestle. Tracking is required only once for training. The artist can then create animated sequences by only defining the (articulated) state of the object. Our models are quite rough, but they suffice for illustration and could be replaced by high quality ones.

The example of the mortar and the pestle is the most basic one, but illustrates well how animated sequences can clarify the intended usage of tools. The animation depicted in

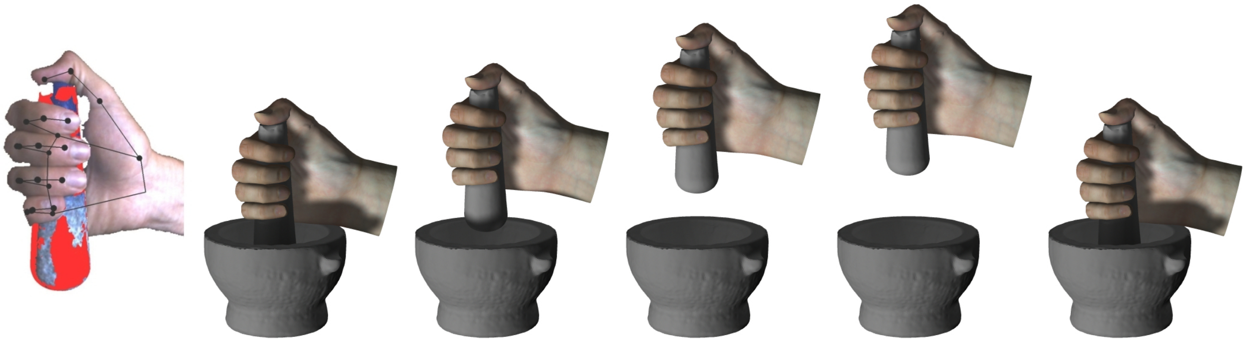


Fig. 10. Generating a sequence with a mortar and a pestle used for crushing. The animation (right) is based on a single observed frame showing a hand holding the pestle (left). The estimated hand pose in that frame is expressed in the coordinate system of the pestle, and the crushing movement of the pestle was defined in Maya.

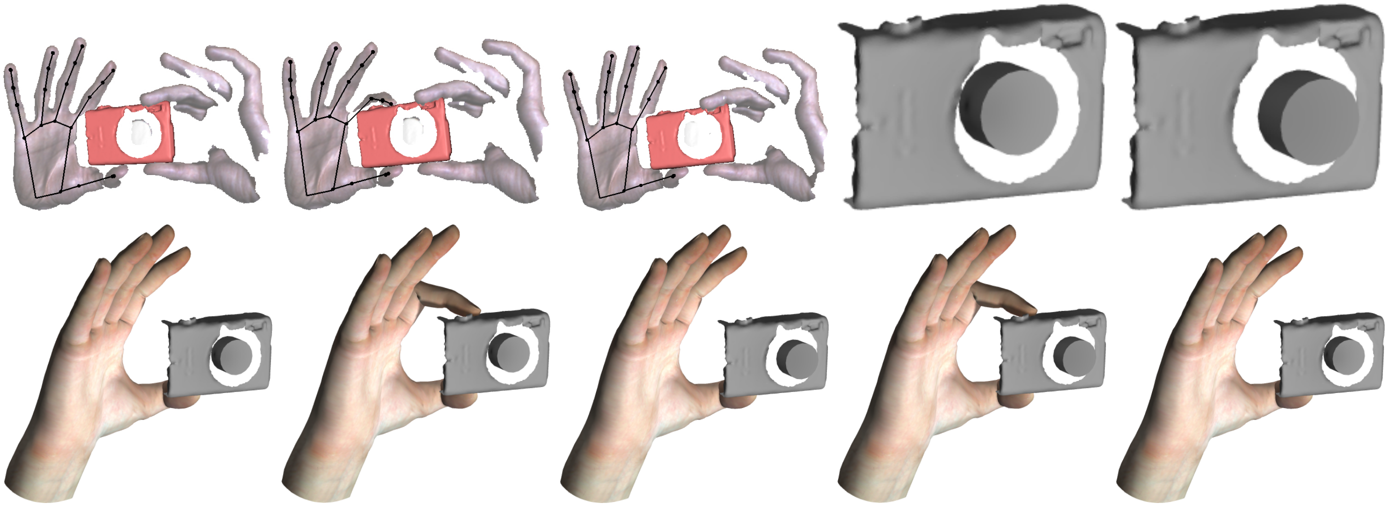


Fig. 11. Generating a sequence involving manipulation of the camera. (top,left) Three frames of an observed sequence in which the hand and the camera were tracked. The estimated hand pose is indicated by the black stick-model skeleton, the partial mesh of the camera registered with the data is drawn in red. In the observed sequence, the lens of the camera emerges and goes back once. (top,right) Close-up of the rendered model of the camera, once with retracted lens and once with emerged lens. (bottom) Frames of the animated sequence. In the complete sequence, the zoom emerges and retracts twice, triggering the respective hand motions with the temporal offset observed in real data.

Fig. 10 (right) is based on a single observed frame showing a hand holding the pestle (see Fig. 10 (left)). The estimated hand pose in that frame is expressed in the coordinate system of the pestle, and the crushing movement of the pestle was defined in Maya. The mortar itself plays only a passive role.

The example of the camera (Fig. 11) is more advanced because the lens can be in or out, and temporal dependencies have to be considered: the index finger approaches the button and starts to flex again *before* the lens comes out. In the tracked sequence (top row, left), the demonstrator presses the button on the camera twice, causing the lens of the camera to emerge and then to retract again. In the object animation created in Maya, the zoom emerges and retracts twice, triggering the respective hand movements to create the final animation (two cycles of the bottom row).

The case of the cup is a little different. Since the cup consists of a single rigid body, the artist can only animate its translation and rotation in Maya. However, to model the grasping process, we augment the cup's state space with a binary flag indicating whether the animated cup is moving or not. When it does move, a firm grasp of the hand on the handle must be established. Consequently, the process of grasping must be initiated *before* the artist wants to change the position of the cup. This temporal offset, the key hand poses, and the hand pose transitions between key poses are again obtained from the observation. Fig. 12 is dedicated to the cup example. In the tracked sequence (top row), the cup is grasped, lifted, put down, and released. In contrast, in the animation (middle row), the cup is not only lifted but also poured out. Two close-ups (bottom row) illustrate this difference. The cup model was created by mirroring the corresponding mesh and has almost no holes.

Finally, we come to the clamshell phone. The artist controls its translation and rotation, as well as the articulated state (phone closed or open). In addition, object contact can be enforced in desired frames in order to let the animated hand dial an arbitrary number. The tracked sequence is shown in the top row of Fig. 13. To track the object, we registered the respective mesh (phone closed or open) with the data. The tracked hand initially holds the closed phone.

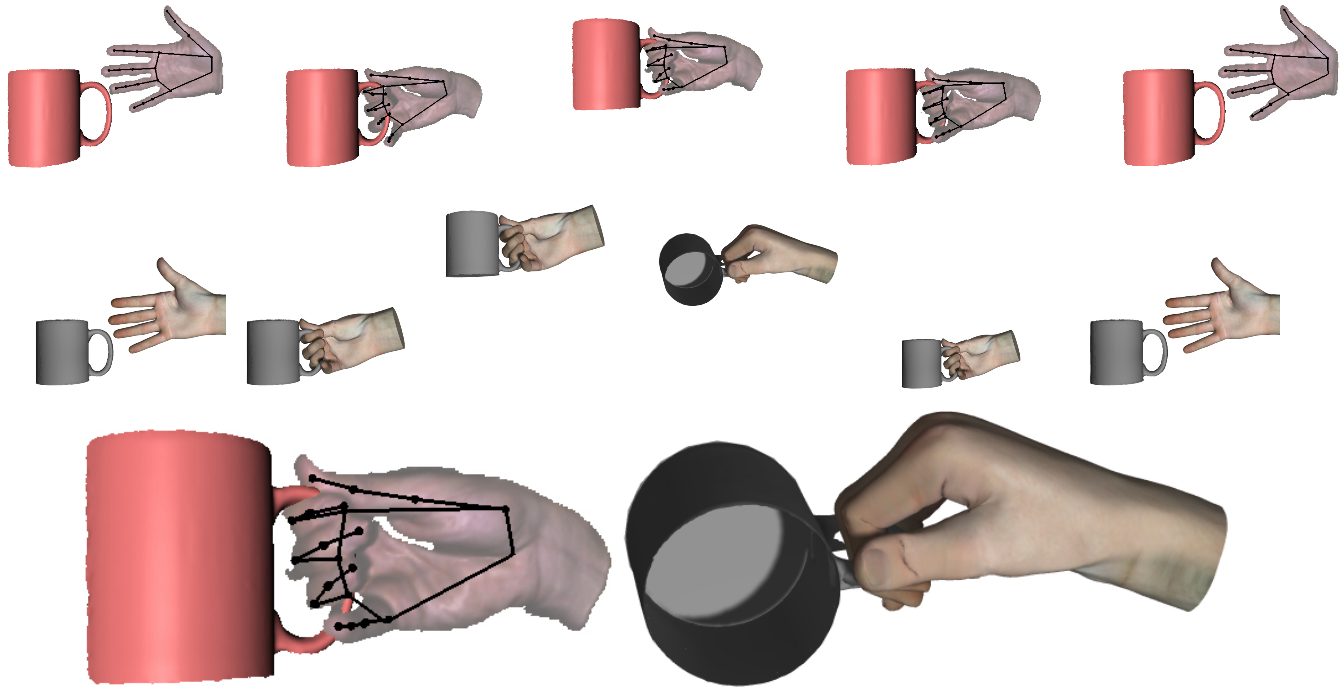


Fig. 12. Generating a sequence involving manipulation of the cup. (top) The tracked sequence. Hand poses are drawn in black, the registered mesh of the cup in red. The cup is grasped, lifted up, put down, and released. No pouring is demonstrated. (middle) An animated sequence in which the cup is not only lifted but also poured. The movement of the cup and the pouring together with the corresponding hand motion results from the object animation in Maya. (bottom) Close-up of one tracked and one animated frame.

The phone is then opened and the digits from one to nine are dialed in order. Thereafter the phone is closed again. In the animation (middle row), the phone is first picked up. This results from a simple rigid transformation of the phone in its closed state. Then, the phone is swung open. In this case the timing of the animation is different than that of the observed demonstration, so the observed hand pose transition has to be stretched. While the phone is open, the animated hand dials a number defined by the artist. Finally, the phone is closed again, and a rigid transformation is applied to lay the phone down. Some texture information was added to the model in Maya. Close-ups are provided in the bottom row.

VI. Conclusions

We presented a data-driven approach for animating object manipulation. While the artist has full control of the object when creating an initial object animation, our approach automatically generates the corresponding hand motion. To this end, we assume that a previously observed manipulation of the object has been captured. Once the data has been processed by our semi-automatic acquisition system and the states of the object have been identified, new animations can be created easily using standard 3d software like Maya. Our current implementation requires that the observed and the animated object are very similar. This, however, could be compensated by acquiring a dataset of objects. Since our model is data-driven and not physical, arbitrary deformable objects cannot be handled. Nevertheless, our experiments have shown that our approach is able to synthesize hand motions that go beyond grasp motions and that involve dynamical changes of the articulated state of an object. Therefore, the proposed method has many applications, e.g., it could be used to create virtual video tutorials demonstrating the usage of tools.

VII. Acknowledgments

The authors gratefully acknowledge support through the EC Integrated Project 3D-Coform.

References

[1] I. Albrecht, J. Haber, and H. Seidel. Construction and animation of anatomically based human hand models. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 98–109, 2003.

[2] A. Bicchi and V. Kumar. Robotic grasping and contact: a review. In *International Conference on Robotics and Automation (ICRA)*, pages 348 – 353, 2000.

[3] M. Cutkosky and P. Wright. Modeling manufacturing grips and correlations with the design of robotic hands. In *International Conference on Robotics and Automation (ICRA)*, pages 1533–1539, 1986.

[4] S. Ekvall and D. Kragíc. Grasp recognition for programming by demonstration. In *International Conference on Robotics and Automation (ICRA)*, pages 748 – 753, 2005.

[5] G. ElKoura and K. Singh. Handrix: animating the human hand. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 110–119, 2003.

[6] J. Gibson. *The ecological approach to visual perception*. Houghton Miffin, Boston, 1979.

[7] H. Hamer, J. Gall, T. Weise, and L. Van Gool. An object-dependent hand pose prior from sparse training data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 671 – 678, 2010.

[8] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool. Tracking a hand manipulating an object. In *International Conference on Computer Vision (ICCV)*, pages 1475–1482, 2009.

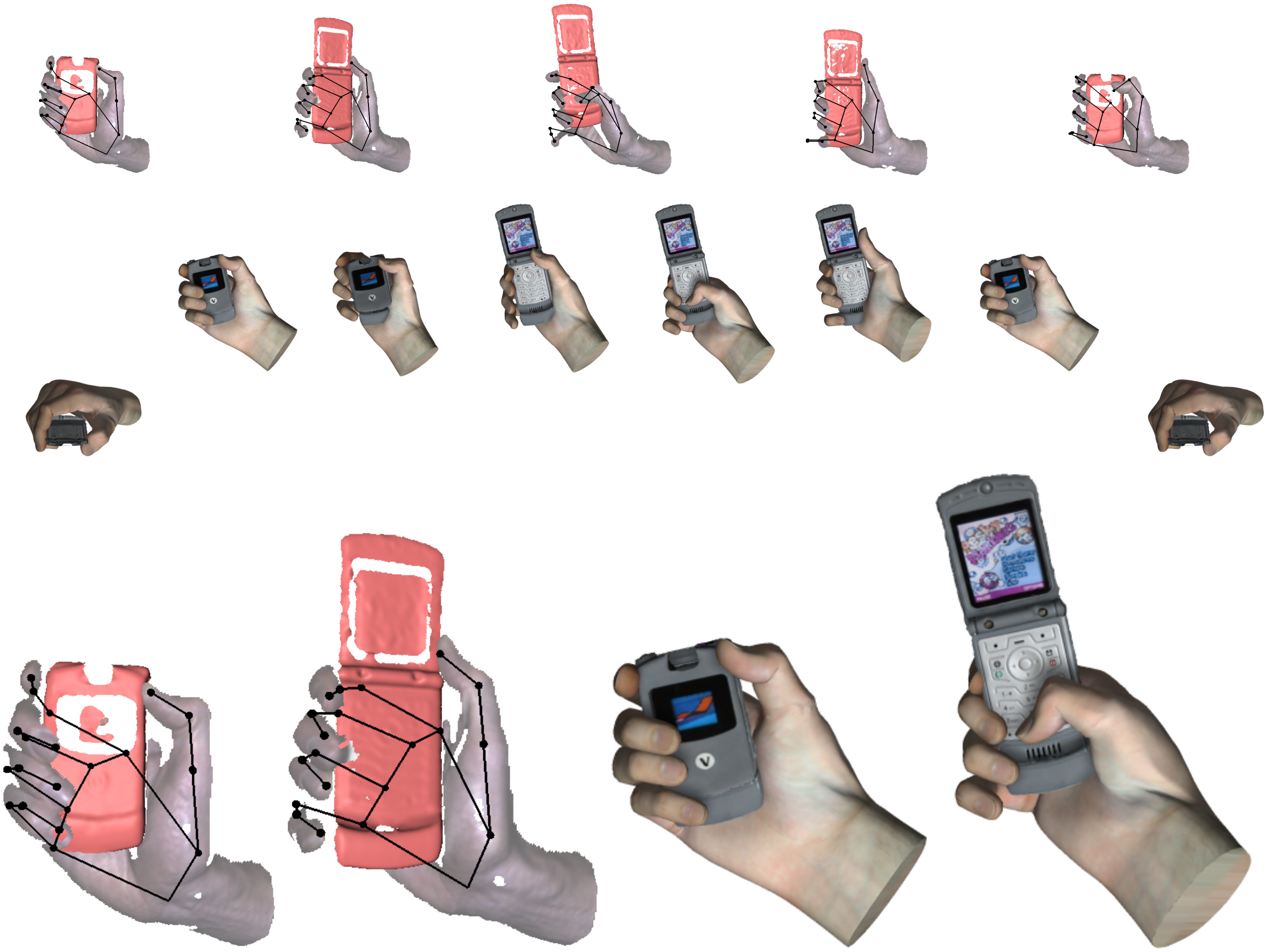


Fig. 13. Generating a sequence involving the clamshell phone. (top) The tracked sequence. Hand poses are drawn in black, the registered mesh of the phone in red. The phone is opened, the key from 1..9,0 are pressed in order, and the phone is closed again. (middle) In the animated sequence the phone is first picked up (which was never observed) and then opened. The thumb movement during opening is interpolated based on the observation, resulting in a kind of flicking motion. After opening the phone, the animation artist can dial an arbitrary number via the definition of contact points. The interpolation between dialing poses is fast in the beginning and slower in the end, to create a realistic impression. Finally, the phone is closed and put down. (bottom) Close-up of some frames.

[9] M. Hueser and T. Baier. Learning of demonstrated grasping skills by stereoscopic tracking of human hand configuration. In *International Conference on Robotics and Automation (ICRA)*, pages 2795–2800, 2006.

[10] K. H. Jo, Y. Kuno, and Y. Shirai. Manipulative hand gesture recognition using task knowledge for human computer interaction. In *International Conference on Automatic Face and Gesture Recognition (FG)*, pages 468–473, 1998.

[11] J. Kim, F. Cordier, and N. Magnenat-Thalmann. Neural network-based violinist's hand animation. In *Computer Graphics International (CGI)*, pages 37 – 41, 2000.

[12] H. Kjellström, J. Romero, D. Martínez, and D. Kragíc. Simultaneous visual recognition of manipulation actions and manipulated objects. In *European Conference on Computer Vision (ECCV)*, pages 336–349, 2008.

[13] P. Kry and D. Pai. Interaction capture and synthesis. *ACM Transactions on Graphics (TOG)*, 25(3):872–880, July 2006.

[14] Y. Li, J. L. Fu, and N. S. Pollard. Data-driven grasp synthesis using shape matching and task-based pruning. *Transactions on Visualization and Computer Graphics (TVCG)*, 13(4):732–747, Aug. 2007.

[15] C. K. Liu. Synthesis of interactive hand manipulation. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 163–17, 2008.

[16] C. K. Liu. Dextrous manipulation from a grasping pose. *ACM Transactions on Graphics (TOG)*, 28(3):1–6, Aug. 2009.

[17] A. Miller, S. Knoop, and H. Christensen. Automatic grasp planning using shape primitives. In *International Conference on Robotics and Automation (ICRA)*, pages 1824 – 1829, 2003.

[18] N. S. Pollard and V. Zordan. Physically based grasping control from example. In *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 311–318. ACM, 2005.

[19] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *International Journal of Computer Vision (IJCV)*, 50(2):203–226, Nov. 2002.

[20] J. Romero, H. Kjellström, and D. Kragíc. Hands in action: real-time 3D reconstruction of hands in interaction with objects. In *International Conference on Robotics and Automation (ICRA)*, pages 458 – 463, 2010.

[21] S. Rusinkiewicz, O. Holt-Hall, and M. Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (TOG)*, 21(3):438–446, July 2002.

[22] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *International Journal of Robotics (IJRR)*, 27(2):157–173, Feb. 2008.

[23] T. Weise, B. Leibe, and L. Van Gool. Fast 3D scanning with automatic motion compensation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2007.