Optimized 4D DPM for Pose Estimation on RGBD Channels using Polisphere Models

Enrique Martinez[1], Oliver Nina[2], Antonio J. Sanchez[1] and Carlos Ricolfe[1]
[1]*Instituto AI2, Universitat Politecnica de Valencia, C/ Vera s/n, Valencia, Spain*
[2]*CRCV, University of Central Florida, Orlando, U.S.A.*
enmarbe1@etsii.upv.es, onina@eecs.ucf.edu, {asanchez, cricolfe}@isa.upv.es

Keywords: Human Pose Estimation, DPM, RGBD Images, Inverse Kinematics.

Abstract: The Deformable Parts Model (DPM) is a standard method to perform human pose estimation on RGB images, 3 channels. Although there has been much work to improve such method, little work has been done on utilizing DPM on other types of imagery such as RGBD data. In this paper, we describe a formulation of the DPM model that makes use of depth information channels in order to improve joint detection and pose estimation using 4 channels. In order to offset the time complexity and overhead added to the model due to extra channels to process, we propose an optimization for the proposed algorithm based on solving direct and inverse kinematic equations, that form we can reduce the interested points reducing, at the same time, the time complexity. Our results show a significant improvement on pose estimation over the standard DPM model on our own RGBD dataset and on the public CAD60 dataset.

1 INTRODUCTION

Human pose estimation is a problem that in recent years has gained much attention. Some of the most well known methods for human pose estimation are based on the Deformable Parts Model (Felzenszwalb et al., 2008; Felzenszwalb et al., 2010; Yang and Ramanan, 2013).

Although, there has been a lot of work on recent years on attempting to improve the DPM model, little work has been done on utilizing the DPM model on 3D vision channels such as RGBD.

In this work, we propose a new formulation for the DPM model that takes advantage of depth information on RGBD images in order to improve the detection of parts of the model as well as the overall human pose estimation.

We also reduce the computational cost of training and testing a DPM model with increased number of channels by a novel approach solving kinematic equations. More specifically, our method treats each part of the body as a semi rigid object and use Denavit-Hartemberg (DH) (Waldron Prof and Schmiedeler Prof, 2008; Khalil and Dombre, 2004) to solve direct and inverse kinematics equations in order to lower the time complexity of our algorithm.

1.1 Background

Felzenszwalb (Felzenszwalb and Huttenlocher, 2005) presented a computationally efficient framework for part-based modeling and recognition using RGB channels. Saffari (Saffari et al., 2009) introduced an on-line random forest algorithm also for pose estimation prediction.

One of the most popular methods for pose estimation prediction was published by Ramanan (Felzenszwalb et al., 2008; Felzenszwalb et al., 2010; Yang and Ramanan, 2013). Ramanan's original model uses a human detection system based on mixtures of multiscale Deformable Parts Model (DPM) using RGB images.

There has been other methods attempting to solve pose estimation such as Wang (Wang et al., 2012) who considers the problem of parsing human poses and recognizing their actions with part-based models introducing hierarchical poselets.

Shotton (Shotton et al., 2013) proposed a method to quickly and accurately predict 3D positions of body joints from a single depth image. Song (Song and Xiao, 2014) proposed to use depth maps for object detection and design a 3D detector to overcome major difficulties of recognition.

In this paper we introduce a novel DPM model base on Ramanan's original method with the goal

Martinez E., Nina O., SÃąnchez A. and Ricolfe C.
Optimized 4D DPM for Pose Estimation on RGBD Channels using Polisphere Models.
DOI: 10.5220/0006133702810288
In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2017)*, pages 281-288
ISBN: 978-989-758-226-4

to take advantage of additional channels such as the depth channel in RGBD data. The intuition behind our approach is that by using depth channels, we leverage additional 3D position information about the objects and the scene in the image. Thus, we obtain a more robust method against image artifacts such as color, illumination, among others.

Furthermore, we propose an optimization for our 4D DPM model that reduces the number of parts to be trained. Thus, we are able to successfully reduce 26 parts from the original DPM model to only 10. This optimization reduces the computational cost of the model while still preserving high degree of accuracy. In the next sections we describe our method in detail and present our results.

2 PROPOSED METHOD

In this section we describe the propoused method in three steps. First we explain our formulation of the Deformable Parts Model to accommodate for additional channels such as the depth channel. Then we talk about the pre-processing step for the Depth channel in which we make a foreground segmentation to improve the accuracy of our algorithm. Finally, we explain the optimization of the computation complexity of our model.

2.1 4D DPM Model

We extend the original DPM model by Ramanan (Felzenszwalb et al., 2010; Yang and Ramanan, 2013) which is used for articulated human detection and human pose estimation by creating a new formulation that extends to alternative channels.

More formally, let us define the score function for a configuration of parts as a sum of local and pairwise scores (co-occurrence model):

$$S(t) = \sum_{i \in V} b_i^{t_i} + \sum_{ij \in E} b_{ij}^{t_i,t_j} \tag{1}$$

where $i \in \{1,...K\}$ and K is the number of parts of the model. $t_i \in \{1,...T\}$ and t_i is the type of part i. $b_i^{t_i}$ is a parameter that favors particular type assignments for part i, while the pairwise parameter $b_{ij}^{t_i,t_j}$ favors particular co-occurrences of part types.

Let us also define $G = (V,E)$ for a K-node relational graph whose edges specify which pairs of parts are constrained to have consistent relations.

Hence the score function associated with a configuration of part types and positions for all RGBD channels is defined as:

$$S(I,p,t) = S(t) + \sum_{i \in V} \begin{bmatrix} \omega_{i\ rgb}^{t_i} \cdot \phi\left(I_{rgb}, p_i\right) \\ \omega_{i\ depth}^{t_i} \cdot \phi\left(I_{depth}, p_i\right) \end{bmatrix} + \\ + \sum_{ij \in E} \begin{bmatrix} \omega_{ij\ rgb}^{t_i,t_j} \cdot \upsilon(p_i - p_j)_{rgb} \\ \omega_{ij\ depth}^{t_i,t_j} \cdot \upsilon(p_i - p_j)_{depth} \end{bmatrix} \tag{2}$$

where $\phi\left(I_{rgb}, p_i\right)$ and $\phi\left(I_{depth}, p_i\right)$ are feature vectors from RGB (I_{rgb}) and Depth (I_{depth}) images respectively and extracted from pixel location p_i. We define $\upsilon(p_i - p_j)_{rgb} = \begin{bmatrix} dx & dx^2 & dy & dy^2 \end{bmatrix}_{rgb}$, where $dx = x_i - x_j$ and $dy = y_i - y_j$, is the relative location of part i with respect to j on image I_{rgb}, and in a similar way for I_{depth}.

The second term in equation 2 is an appearance model that computes the local score of placing a template $\omega_{i\ rgb}^{t_i}$ or $\omega_{i\ depth}^{t_i}$ for part i, tuned for type t_i at location p_i.

The third term in equation 2 can be interpreted as a switching spring model that controls the relative placement of part i and j by switching between a collection of springs. Each spring is tailored for a particular pair of types (t_i, t_j), and is parameterized by its rest location and rigidity, which are encoded by $\omega_{i\ rgb}^{t_i}$ or $\omega_{i\ depth}^{t_i}$ depending on which image is used.

The goal is to maximize $S(x,p,t)$ from the above formulation over p and t. When the relational graph $G = (V,E)$ is a tree, this can be done efficiently with dynamic programing in the following way. Let $kids(i)$ be the set of children of part i in G. We compute the message part i that passes to its parent j in this way:

$$score_i(t_i,p_i) = b_i^{t_i} + \begin{bmatrix} \omega_{t_i\,rgb}^{i} \cdot \phi\left(I_{rgb}, p_i\right) \\ \omega_{t_i\,depth}^{i} \cdot \phi\left(I_{depth}, p_i\right) \end{bmatrix} + \\ + \sum_{k \in kids(i)} m_k(t_i,p_i) \tag{3}$$

$$m_i(t_j,p_j) = \max_{t_i} b_{ij}^{t_i,t_j} + \max_{p_i} score(t_i,p_i) + \\ + \begin{bmatrix} \omega_{ij\ rgb}^{t_i,t_j} \cdot \upsilon(p_i - p_j)_{rgb} \\ \omega_{ij\ depth}^{t_i,t_j} \cdot \upsilon(p_i - p_j)_{depth} \end{bmatrix} \tag{4}$$

Equation 3 computes the local score of part i, at all pixel locations p_i and for all possible types t_i, by collecting messages from the children of i. Equation 4 computes every location and type of its child part i. Once messages are passed to the root part $(i = 1)$, $score_1(c_1,p_1)$ represents the best scoring configuration for each root position and type.

Figure 1 shows our 4D DPM model learned with only 10 parts, trained on our dataset. We show the

local templates in Figure 1 part (a), and the tree structure in Figure 1 part (b), placing parts at their best-scoring location relative to their parent.

Though we visualize 4 trees generated by selecting one of the four types of each part, and placing it at its maximum-scoring position, there exists an exponential number of possible combinations, obtained by composing different part types together. Notice that the standard DPM model consists of 14 or 26 parts which are used on RGB images only. In our case we only need 10 parts to train and test our model. The reason for this is to reduce the computational complexity of the model which we explain in more detail in section 2.3.

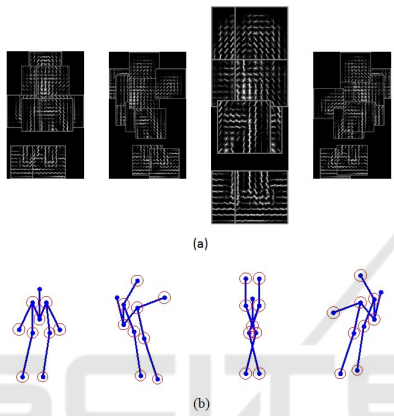

Figure 1: Our Learned Model: 10 parts using our dataset. (a): the local templates. (b): the tree structure.

2.2 Foreground Segmentation

Although D channels in RGBD data provide important information about location of the target object, it is still necessary to make a foreground segmentation from the target object. By making the foreground segmentation, we accent the image contrast between foreground and background as well as remove any noise that could negatively affect our model.

In order to automatically make a foreground segmentation from depth channels, we use a Maximally Stable Extremal Regions (MSER) based approach (Matas et al., 2004). MSER regions are regions that are most stable through a range of all possible threshold values applied to them. More formally:

Given a seed pixel x, and a parameter Δ which represents the intensity variation in the scale of x, we define the stability property S of a region R as:

$$S = \frac{|\Delta R - R|}{|R|} \tag{5}$$

where the unary operator $||$ represents the area of the region input. Hence, MSER regions are those with higher S values.

Given a Depth channel image, we use equation 5 to obtain the most stable regions from the channel. We then remove those MSRE regions that hold the following property

$$|R| > T \tag{6}$$

where T is a certain threshold for the area of the region. We can see in Figure 2 the results from our segmentation method.

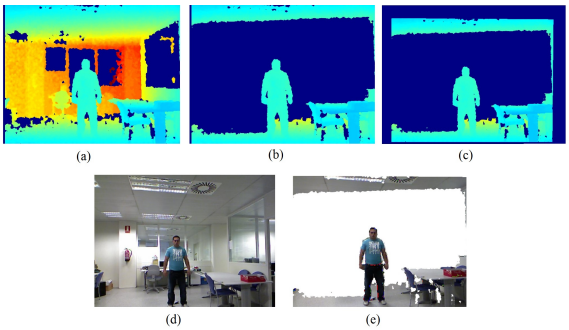

Figure 2: (a) Original Depth, (b) Depth after applying MSER; (c) Conversion of Depth pixel position to RGB pixel position; (d) Original RGB; (e) Combining image (c) and (d).

2.3 Model Optimization

The additional Depth channel included in proposed formulation of DPM adds extra computational cost that makes training and testing cumbersome.

In this section we explain an optimization technique that makes use of inverse kinematic equations in order to infer other parts by training with fewer ones. We will first describe the system and calibration process we use in order to use the propoused optimization step.

2.3.1 3D Vision System

The proposed vision system used in our experiments is the Kinect camera, which consists of two optical sensors whose interaction allows a three-dimensional scene analysis.

One of the sensors is a RGB camera which has a video resolution of 30 fps. The image resolution given by this camera is 640x480 pixels. The other sensor is an infrared camera that gathers depth information from objects found in the scene.

The main purpose of this sensor is the emission of an infrared signal which is reflected off of objects being visualized and then recaptured by a monochrome CMOS sensor. A matrix is then obtained which provides a depth image of the objects in the scene. Hence, calibration at this stage is much needed to correlate both camera and world coordinate system.

2.3.2 3D Vision System Calibration

The intrinsic and extrinsic parameters of the two Kinect optical sensors are different. Therefore, it is necessary to calibrate one optical sensor (RGB) with respect to the other (Depth) in order to correlate the corresponding pixels in both images. The calibration system is done in a similar way to (Berti et al., 2012) or (Viala et al., 2011) and (Viala et al., 2012).

Using RGBD information together with a calibration system, we can know where a pixel is on an image with respect to the camera, we can also convert a point in pixel coordinates (x_{pixel}, y_{pixel}) (2D coordinates inside the image) to another point in camera coordinates $(x_{camera}, y_{camera}, z_{camera})$ (3D coordinates of one point in the world with respect to the camera). Thus, we use the point in camera coordinates to calculate the kinematic equations.

2.4 Model of Human Body using Polispheres

In order to track the human skeleton, it is necessary to perform the modeling of the human body. The human body is modeled as a set of articulated rigid structures to perform the necessary simplifications and to obtain a simple model. For each of these articulated rigid structures the state variables and kinematics are obtained. Kinematics are calculated using DH (Denavit-Hartemberg).

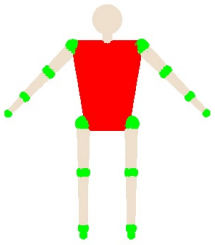

Figure 3: Body human model.

Figure 3 shows the geometrical model used, where the green spheres indicate the main areas to represent each of the limbs.

For use polysphere, we have modeled each part of the body using two points, between these two points we have one number defined of spheres, the first and the last one represents our articulations. The body are modeled using 4 points (hips and shoulders). Figure 4 shows a representation of polisphere model used.

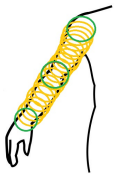

Figure 4: Polisphere representation. Green: sphere centered on articulated join. Yellow: spheres encompassing the shaped part of the body.

2.5 State Variable

The human body is designed as if it were a set of articulated rigid structures performing collision detection. More concretely, our human body model is composed of 4 different articulated rigid structures: 1 structure for each arm and 1 structure for each leg. Also, state variables for each of these articulated rigid structures are created.

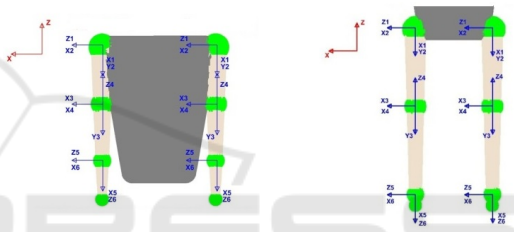

Figure 5: State Variable. Left: arms, Right: legs.

2.5.1 Denavit Hartemberg (DH)

To control each of these articulated rigid structures to which human body model has been reduced to, we use DH (Waldron Prof and Schmiedeler Prof, 2008; Khalil and Dombre, 2004). We use 6 joints for each articulated rigid structure, starting with shoulders or hips and ending with hands or feet respectively.

First, we establish the base coordinate system (X_0, Y_0, Z_0) at the supporting base with Z_0 axis lying along the axis of motion of joint 1. We establish a joint axis and align the Z_i with the axis of motion of joint $i+1$. We also locate the origin of the i_{th} coordinate at the intersection of the Z_i and Z_{i-1} or at the intersection of a common normal between the Z_i and Z_{i-1}. Then, we establish $X_i = \pm(Z_{i-1} \times Z_i) / \|Z_{i-1} \times Z_i\|$ or along the common normal between the Z_i and Z_{i-1} axes when they are parallel. We also assign Y_i to complete the right-handed coordinate system. Finally, we find the link and joint parameters: θ_i, d_i, a_i, α_i. Figure 5 shows our kinematic model.

We can use this information to calculate the inverse and direct kinematics. For direct kinematics, given the 6 variable joints $(q_1, q_2, q_3, q_4, q_5, q_6)$,

we obtain the coordinates of end effector (x,y,z) with respect the base of the articulated rigid structure. For inverse kinematics, given the coordinates of end effector and the orientation in euler parameters, (x,y,z,ϕ,θ,ψ), we obtain the 6 variable joints, $(q_1,q_2,q_3,q_4,q_5,q_6)$.

Using the information above, we calculate direct kinematics using a homogeneous transformation matrix:

$$^{i-1}A_i(q_i) = \quad (7)$$
$$\begin{bmatrix} c(\theta_i) & -c(\alpha_i)\cdot s(\theta_i) & s(\alpha_i)\cdot s(\theta_i) & a_i\cdot c(\theta_i) \\ s(\theta_i) & c(\alpha_i)\cdot c(\theta_i) & -s(\alpha_i)\cdot c(\theta_i) & a_i\cdot s(\theta_i) \\ 0 & s(\alpha_i) & c(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Equation 7 relates a joint with the previous. We need identify the location of the end effector relative to the reference, equation 8 show the matrix necessary to do it:

$$^0T_6(q_1,q_2,q_3,q_4,q_5,q_6) = {}^0A_1(q_1)\cdot {}^1A_2(q_2)\cdot \cdot {}^2A_3(q_3)\cdot {}^3A_4(q_4)\cdot {}^4A_5(q_5)\cdot {}^5A_6(q_6) \quad (8)$$

To calculate inverse kinematics is necessary use geometric models for the first three joints because it can not be done using the inverse transform technique. We have the coordinates for the final effector (x,y,x) and after apply geometric models we obtain the first three joints:

$$q_1 = \arctan\left(\frac{y}{x}\right) \quad (9)$$

$$q_3 = \arctan\left(\frac{\pm\sqrt{1-\cos^2\left(\frac{x^2+y^2+z^2-a_2-a_3}{2\cdot a_2\cdot a_3}\right)}}{\cos\left(\frac{x^2+y^2+z^2-a_2-a_3}{2\cdot a_2\cdot a_3}\right)}\right) \quad (10)$$

$$q_2 = \arctan\left(\frac{z}{\pm\sqrt{x^2+y^2}}\right) - \quad (11)$$
$$-\arctan\left(\frac{a_3\cdot\sin\left(\frac{x^2+y^2+z^2-a_2-a_3}{2\cdot a_2\cdot a_3}\right)}{a_2+a_3\cdot\cos\left(\frac{x^2+y^2+z^2-a_2-a_3}{2\cdot a_2\cdot a_3}\right)}\right)$$

Now we can use inverse kinematics to calculate the last three joints. We write $^0R_6 = [n\ o\ a] = {}^0R_3\cdot {}^3R_6$ for the sub matrix rotation of $^0T_6$. We know $^0R_6$ because is the orientation of the final effector and $^0R_3$ because is defined by $^0R_3 = {}^0R_1\cdot {}^1R_2\cdot {}^2R_3$ using (q_1,q_2,q_3). Then we need calculate:

$$^3R_6 = [r_{ij}] = \left({}^0R_3\right)^{-1}\ {}^0R_6 \quad (12)$$

Applying $^3R_6 = {}^3R_4\cdot {}^4R_5\cdot {}^5R_6$ using (q_4,q_5,q_6), we obtain equation 13.

$$^3R_6 = \quad (13)$$
$$\begin{bmatrix} a+b & a-b & -c(q_4)s(q_5) \\ c-e & d+f & -s(q_4)s(q_5) \\ c(q_6)s(q_5) & s(q_6)s(q_5) & c(q_5) \end{bmatrix}$$

$$Where:$$
$$a = c(q_4)c(q_5)c(q_6) \qquad b = s(q_4)s(q_6)$$
$$c = s(q_4)c(q_5)c(q_6) \qquad d = s(q_4)c(q_5)s(q_6)$$
$$e = c(q_4)s(q_6) \qquad f = c(q_4)c(q_6)$$

We obtain the last three joints using equation 12 and equation 13:

$$q_4 = \arctan\left(\frac{r_{23}}{r_{13}}\right) \quad (14)$$

$$q_5 = \arccos(-r_{33}) \quad (15)$$

$$q_6 = \frac{\pi}{2} - \arctan\left(\frac{r_{32}}{r_{31}}\right) \quad (16)$$

In our case, we use inverse kinematics because we can obtain where the base of our articulated rigid structure (shoulders or hips) is, and where the final effector and the orientation (hands an feet) are, thus we have these parameters: (x,y,z,ϕ,θ,ψ) and using inverse kinematics, we obtain the 6 variable joints,$(q_1,q_2,q_3,q_4,q_5,q_6)$, and use them to know where the elbow or knee is located. Algorithm 1 shows the steps of our method.

Algorithm 1: Algorithm used.

```
Data: RGB and Depth image.
Result: Human body pose estimation.
Initialization.
Load model trained.
Load frames.
for i = 1 : nFrame do
    Read RGB and Depth.
    Apply MSER.
    Convert pixel depth to pixel RGB.
    Obtain points from DPM.
    Convert points to 3D coordinates.
    Apply DH.
    Visualization.
end for
Obtain APK and PCK acuracy.
Obtain error metrics.
```

3 RESULTS

For the evaluation of our results, we use a similar method to (Yang and Ramanan, 2013), where a PCK and APK metrics are used. In PCK we use test images

with tightly-cropped bounding box for each person. Given the bounding box, a pose estimation algorithm reports back keypoint locations for body joints.

A candidate keypoint is defined to be correct if it falls within $\alpha \cdot \max(h, \omega)$ pixels of the ground-truth keypoint, where h and ω are the height and width of the bounding box respectively, and α controls the relative threshold for considering correctness.

For the APK metric, is not necessary the access to the bounding box. One can combine the two problems by thinking of body parts as objects to be detected, and evaluate object detection accuracy with a precision-recall curve (Everingham et al., 2010).

We consider a candidate key point to be correct if it lies within $\alpha \cdot \max(h, \omega)$ pixels of the ground-truth. Thus, we obtain values in the range of 0% and 100%; Hence the higher the value, the better the accuracy.

The second method consists in directly calculating the distance between the results and the correct labeled point. To do this, we use a set of images where all joints have been labeled. The distance between the result and the correct location label represents an error score. For each joint we obtain an error score which is the mean value calculated from all frames.

3.1 Quantitative Results

3.1.1 Foreground Segmentation Evaluation

For our foreground segmentation evaluation method, we use in all cases 6 mixtures parts. The size of the images are 320x240.

The results presented here evaluate the relevance of using our foreground segmentation method. Table 1 shows the results of using foreground segmentation in either the testing or training phases.

Table 1: APK, PCK and Error Metrics For Foreground Segmentation.

Model	B. training	B. testing	keypoint	head	shoulder	wrist	hip	ankle	mean
1	no	no	APK	100	100	89.64	100	100	97.92
			PCK	100	100	92.42	100	100	98.48
			error	4.22	3.66	7.63	5.96	4.43	5.18
2	no	yes	APK	100	100	83.79	100	100	96.75
			PCK	100	100	89.39	100	100	97.87
			error	4.54	**3.61**	7.70	**3.35**	3.77	4.59
3	yes	no	APK	100	100	82.40	100	100	96.49
			PCK	100	100	87.37	100	100	97.47
			error	3.03	4.49	9.65	3.38	3.09	4.72
4	**yes**	**yes**	**APK**	100	100	**95.63**	100	100	**99.12**
			PCK	100	100	**96.46**	100	100	**99.29**
			error	**2.55**	4.70	**5.62**	3.41	**2.64**	**3.78**

"B. training" represents the use of foreground segmentation on the training image and "B. testing" represents the use a foreground segmentation on the test image.

Table 1 shows that our method with foreground segmentation, obtains accuracies of around 99%. We can corroborate these results with our second evaluation, where we obtain the distance between two points. In this case, Table 1 shows our method achieves an average error of only 3.78 compared to 5.18, where no foreground segmentation is performed.

3.1.2 Complete Model Evaluation

3.1.3 Our Dataset

Our dataset consists of 7 videos with only one person on the scene moving his arms and legs. In total we have around 1000 images were people are moving their arms and legs. All of these images are in the same scene but with different objects and different clothes.

We compare our method which uses 6 mixture parts and 10 parts, with the original DPM model which uses 6 mixture parts and 26 parts. We train both models with the same training samples from our own dataset.

Table 2: APK, PCK and Error Metrics For Complete Method.

Model	keypoint	head	shoulder	wrist	hip	ankle	mean
DPM	APK	100	100	78.04	100	100	95.60
	PCK	100	100	83.33	100	100	96.66
	error	4.48	6.29	18.53	3.94	4.25	7.49
Ours	**APK**	100	100	**95.63**	100	100	**99.12**
	PCK	100	100	**96.46**	100	100	**99.29**
	error	**2.55**	**4.70**	**5.62**	**3.41**	**2.64**	**3.78**

We can observe in Table 2 that our method outperforms the standard DPM model, specially on the wrist part with about 3% better accuracy. We can also see that our method achieves lower error rates in all the parts compared to the standard DPM.

3.1.4 CAD60 Dataset

The CAD60 dataset contains 60 RGB-D videos, 4 subjects (two male, two female), 5 different environments (office, bedroom, bathroom and living room) and 12 activities (rinsing mouth, brushing teeth, wearing contact lens, talking on the phone, drinking water, opening pill container, cooking (chopping), cooking (stirring), talking on couch, relaxing on couch, writing on whiteboard, working on computer). The dataset also provides ground truth for the joints of the skeletons that belong to the subjects in the videos.

The CAD60 dataset has been used succesfully for activity recognition as in (Ni et al., 2013; Gupta et al., 2013; Wang et al., 2014; Shan and Akella, 2014; R. Faria et al., 2014). However, our problem is not activity recognition but human pose estimation. Nevertheless, because ground truth for the joints are provided with CAD60, this dataset fulfills our needs well.

Because there are not many publicly available RGBD datasets that provide ground truth joints of subjects, we are limited to use CAD60. Also, to our knowledge, there is no published results on human pose estimation for this dataset. Because of this and because there are not many non-commercial and publicly available methods that deal with RGBD data for pose estimation, we compare our method only to the original DPM model.

For these experiments, our model is trained using the samples from CAD60 dataset. We compare our method to a previously trained DPM model and another DPM model trained solely on CAD60 samples (DPM-t).

Table 3: APK and PCK metrics on the CAD60 dataset.

Model	keypoint	head	shoulder	wrist	hip	ankle	mean
DPM	APK	47.42	66.69	22.95	45.98	47.10	46.02
	PCK	62.00	70.50	39.00	60.00	57.50	57.80
	error	17.35	14.10	35.89	7.06	19.57	18.79
DPM-t	APK	73.02	73.53	32.26	66.33	42.38	57.50
	PCK	78.50	78.50	44.50	70.50	49.50	64.30
	error	15.21	12.30	31.02	6.64	16.31	16.29
P. Method	**APK**	**91.23**	**87.06**	**51.63**	**86.21**	**82.01**	**79.63**
	PCK	**92.80**	**90.00**	**66.00**	**89.00**	**90.00**	**85.56**
	error	**8.81**	**7.53**	**19.25**	**6.05**	**9.25**	**10.17**

Table 3 shows our method outperforms the standard DPM model by a large margin of about 20% accuracy. Table 3 also show the errors rates between correct points and points detected. We can observe that using our model we have around 10 points less of error rate than the standard DPM model.

3.2 Qualitative Results

In this section we analyze the qualitative results of our method. Figure 6 shows different frames where the original DPM model fails, and where our model works better on our dataset. Figure 7 shows different frames where the original DPM model trained with CAD60 dataset (DPM-t) fails, and where our model works better on CAD60. The images in the first row in Figure 6 and Figure 7 show the original DPM model and the images in the second row represent ours. Notice that our model more accurately predicts the pose of the person in the video.

Figure 8 shows the human model obtained through

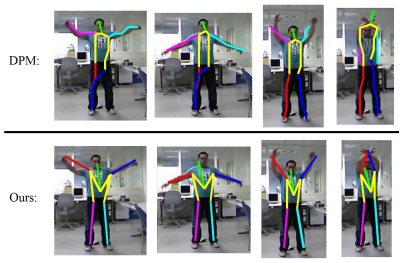

Figure 6: Qualitative comparison of DPM and our method on our proposed dataset.

Figure 7: Qualitative comparison of DPM trained with CAD60 dataset (DPM-t) and our proposed method on CAD60.

DH in different frames. Our kinematic model correctly infers the parts and joints of the human body. We obtain the solutions in Figure 8 applying DH at points obtained through our model.

3.3 Time Complexity Analysis

Here we describe the computational cost between the original DPM model and ours. For our experiments, we use a windows 7 system with 4 GB RAM. We take 99 frames and we calculate for each frame the original DPM model and our model, finally we take the average time between all frames.

In both cases we use the same size of the image, 320x240. Using the original model we have a computational cost of 9.21 seconds for each frame whereas using our model, the computational cost is reduced to 7.26 seconds even though we are processing more channels than the original model. This is roughly a 20% gain in performance.

4 CONCLUSIONS

In this paper, we extend a DPM model that takes advantage of Depth information on RGBD images in order to improve detection of parts and human pose estimation.

We also propose a novel foreground segmentation

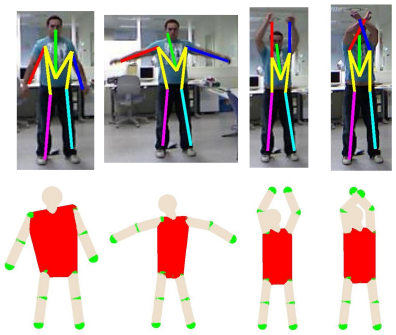

Figure 8: Body human model calculated after obtain the points by our proposed DPM model.

technique ideal for Depth channels of RGBD data that helps us improve our results further.

Finally, we reduce the computational cost of our new DPM model by a novel approach solving kinematic equations. Our results show significant results over the standard DPM model in our dataset and in the publicly available CAD60 dataset.

ACKNOWLEDGEMENTS

This work was partially financed by Plan Nacional de I+D, Comision Interministerial de Ciencia y Tecnologa (FEDER-CICYT) under the project DPI2013-44227-R.

REFERENCES

Berti, E. M., Salmerón, A. J. S., and Benimeli, F. (2012). Human–robot interaction and tracking using low cost 3d vision systems. *Romanian Journal of Technical Sciences - Applied Mechanics*, 7(2):1–15.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79.

Gupta, R., Chia, A. Y.-S., and Rajan, D. (2013). Human activities recognition using depth images. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 283–292. ACM.

Khalil, W. and Dombre, E. (2004). *Modeling, identification and control of robots*. Butterworth-Heinemann.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767.

Ni, B., Pei, Y., Moulin, P., and Yan, S. (2013). Multilevel depth and image fusion for human activity detection. *Cybernetics, IEEE Transactions on*, 43(5):1383–1394.

R. Faria, D., Premebida, C., and Nunes, U. (2014). A probalistic approach for human everyday activities recognition using body motion from rgb-d images. *IEEE RO-MAN'14: IEEE International Symposium on Robot and Human Interactive Communication*.

Saffari, A., Leistner, C., Santner, J., Godec, M., and Bischof, H. (2009). On-line random forests. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1393–1400. IEEE.

Shan, J. and Akella, S. (2014). 3d human action segmentation and recognition using pose kinetic energy. *IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO)*.

Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., and Moore, R. (2013). Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124.

Song, S. and Xiao, J. (2014). Sliding shapes for 3d object detection in depth images. In *Computer Vision–ECCV 2014*, pages 634–651. Springer.

Viala, C. R., Salmeron, A. J. S., and Martinez-Berti, E. (2011). Calibration of a wide angle stereoscopic system. *OPTICS LETTERS, ISSN 0146-9592, pag 3064-3067.*

Viala, C. R., Salmeron, A. J. S., and Martinez-Berti, E. (2012). Accurate calibration with highly distorted images. *APPLIED OPTICS, ISSN 0003-6935, pag 89-101*.

Waldron Prof, K. and Schmiedeler Prof, J. (2008). *Kinematics*. Springer Berlin Heidelberg.

Wang, J., Liu, Z., and Wu, Y. (2014). Learning actionlet ensemble for 3d human action recognition. In *Human Action Recognition with Depth Cameras*, pages 11–40. Springer.

Wang, Y., Tran, D., Liao, Z., and Forsyth, D. (2012). Discriminative hierarchical part-based models for human parsing and action recognition. *The Journal of Machine Learning Research*, 13(1):3075–3102.

Yang, Y. and Ramanan, D. (2013). Articulated human detection with flexible mixtures of parts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(12):2878–2890.