Présentation d'article: Algorithmes en Science des Données

Real-Time Seamless Single Shot 6D Object Pose Prediction

Bugra Tekin - EPFL Sudipta N. Sinha - Microsoft Research Pascal Fua - EPFL

CentraleSupélec

Herbecq Simon
Haumesser Pierrick
Promo SDI 2021

Plan

1. Introduction
2. Etat de l’art
3. Description du modèle
4. Détails sur la mise en oeuvre
5. Description des expériences
6. Résultats et performances
7. Conclusion

Introduction

Objectif: Estimer la pose 6D d'un objet à partir d'une image 2D en temps réel

L'estimation de la pose 6D est crucial dans des domaines comme la RA, la VR ou la robotique

Méthodes habituelles:

- 1ère étape: estimation
- 2ème étape: post-traitement des résultats
- Nécessite les 2 étapes pour être suffisamment performant
- Pas idéal pour les applications en temps réel, peut traiter 10 images par secondes
- Temps de calcul augmente si plusieurs objets

Méthode proposée:

- Une seule étape: Détecte la position d'un objet sur une image et prédit sa pose 6D (position et orientation) dans l'environnement
- Bonne performance avec une seule étape
- Rapide, peut traiter 50 images par secondes
- Pas de d'augmentation de temps de calcul si plusieurs objets
- Astuce: prédire la projection des sommets de la boîte englobante 3D sur l'image 2D avec un CNN puis estimer la pose 6D avec un algorithme PnP Perspective-n-Point

Etat de l’art

Méthodes classiques:

- Utilisent images RGB classiques
- Estiment pose à l’aide de descripteurs locaux
- Rapide et robuste à l’occlusion
- Nécessitent des objets texturés et une image en haute résolution, ce qui n’est pas souvent le cas dans des environnements réels avec des caméras mobiles
- Exemples de méthodes:
  - Hausdorff matching
  - Oriented Chamfer matching for edges
  - 3D Chamfer matching for aligning 3D curve-based models to images

Méthodes RGB-D:

- Méthodes robustes qui exploitent un capteur de profondeur pour améliorer les résultats
- Capteur de profondeur souvent trop coûteux en énergie pour les caméras mobiles
- Exemples de scientifiques ayant travaillé sur ces méthodes:
  - Hinterstoisser
  - Rios et al.
  - Brachmann et al.
  - Zach et al.

Etat de l’art

Méthodes Deep-Learning avec CNN:

- 1) Détection Pose 6D
  - CNN dominent depuis quelques années dans le domaine de l’estimation de pose
  - Nécessitent souvent un post-traitement, les rendant lentes surtout avec plusieurs objets à détecter
  - peut être traité comme de la classification en discrétisant l’espace des poses
  - peut être traité comme de la régression:
    - PoseNet
      - Estime la composante de transition et de la composante de rotation.
      - Une unique Loss pour les deux composantes, difficile à paramétrer
    - PoseCNN cherche à dépasser ce problème
      - Les deux estimateurs sont complètement découplés
      - Deux Loss distinctes.
  - Nouvelles méthodes:
    - **estimer des coordonnées 2D (boîte englobante) et en déduire la pose 6D**
    - Plus de problèmes de loss car les prédictions se font sur une image 2D
    - Les modèles sont plus stables et obtiennent donc de meilleures résultats
    - C’est l’approche adoptée dans cette article.

Etat de l’art

Méthodes Deep-Learning avec CNN:

- 2) Détection d’objet en 2D:
  - **Méthodes “single-shot CNN” apparaissent**
    - ne nécessitent pas de post-traitement
    - Exemples : Faster-RCNN, YOLO, SSD
    - l’article veut étendre ces méthodes à la détection de pose 6D d’objets.

Description du modèle

- **YOLO:**
  - un single-shot détecteur d’objet en 2D
  - permet à l'origine de trouver la boîte englobante 2D d’un objet

- **Le modèle proposé:**
  - s’inspire de YOLO
  - conçu pour prédire la projection sur l’image 2D de la boîte englobante 3D
  - Après modification de YOLO, le CNN estime ces coordonnées 2D. La pose 6D peut être alors calculée algébriquement avec un algorithme PnP efficace.

Description du modèle

- **Description des objets:**
  - 9 points de contrôles:
    - 8 points pour les sommets de la boîte englobante 3D
    - 1 point pour le centre de l'objet
  - Description classique, qui fait sens pour la plupart des objets et qui assure une bonne répartition des points sur l'image 2D
- **Entrée:** Une image RGB
- **Structure:** Fully-convolutional architecture
  - Divise l'image 2D en une grille régulière SxS, avec S=13
  - 23 convolutional layers - 5 max-pooling layers
  - 1 Passthrough layer permettant de passer des informations provenant d'un layer précédent
- **Output:** SxSxD 3D tensor
  - D = 9x2+C+1
    - 9 (xi, yi) points de contrôles
    - C probabilités de classe
    - 1 valeur de confiance

Description du modèle

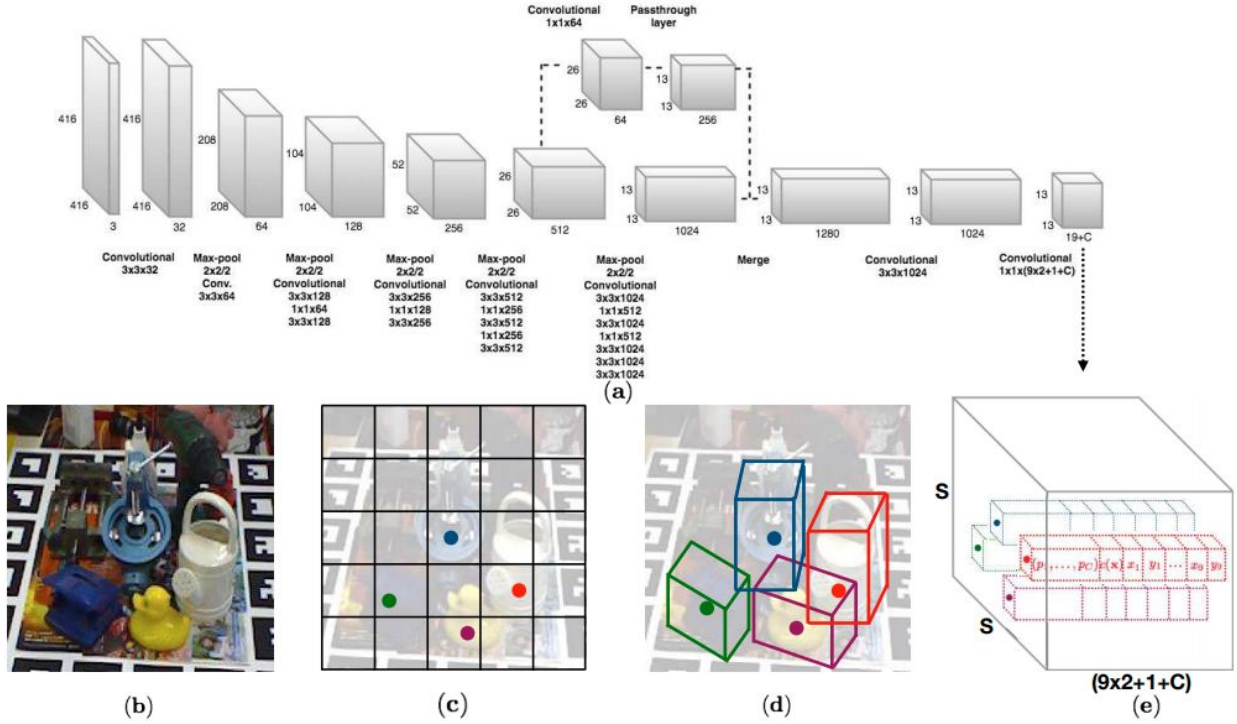

Figure 1: Overview: (a) The proposed CNN architecture. (b) An example input image with four objects. (c) The $S \times S$ grid showing cells responsible for detecting the four objects. (d) Each cell predicts 2D locations of the corners of the projected 3D bounding boxes in the image. (e) The 3D output tensor from our network, which represents for each cell a vector consisting of the 2D corner locations, the class probabilities and a confidence value associated with the prediction.

Description du modèle

Le réseau est entraîné à prédire les **9 points de contrôles**

Il est important que les valeurs soient prédites avec **une grande confiance dans les zones avec un objet, et une faible confiance dans les zones sans objets.**
Pour obtenir ces confiances, la méthode initiale de YOLO n'est pas applicable car trop lente pour le cas traité
Approche utilisée: modéliser la confiance avec la fonction figure 2

DT(x) est la distance euclidienne sur l'image. La finesse de l'exponentielle est réglée avec α.

Cette fonction permet de mesurer la différence entre les coordonnées prédites et la vérité terrain.

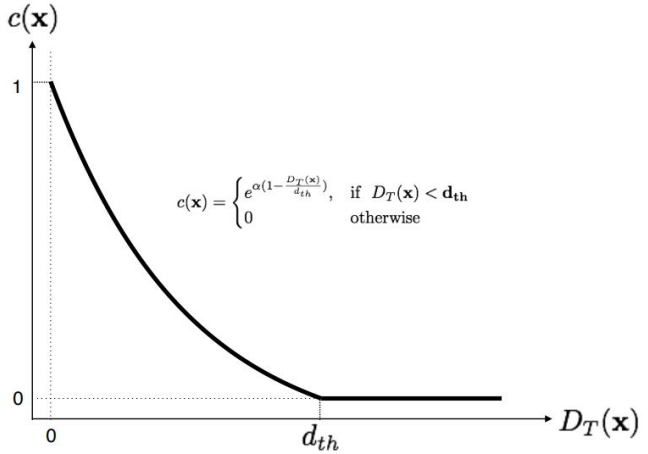

Figure 2: Confidence $c(\mathbf{x})$ as a function of the distance $D_T(\mathbf{x})$ between a predicted point and the true point.

Description du modèle

- **Entrainement**
  - Ne prédit pas vraiment les coordonnées (x, y), mais les coordonnées f(x) et f(y) relatives aux coins en haut à gauche de la cellule courante (cx, cy).
    Nous obtenons donc les coordonnées réel (gx, gy) ainsi: $g_x = f(x) + c_x$ $g_y = f(y) + c_y$
    - Pour le centroid, f(x) et f(y) sont compris entre 0 et 1 car le centre doit-être inclus dans la cellule (sigmoid)
    - Pas de contrainte pour les sommets de la boîte englobante qui peut sortir de la cellule (identité)
    - Cela force le réseau à d'abord trouver la cellule approximative de l'objet puis à calculer les sommets
  - Loss function à minimiser: $\mathcal{L} = \lambda_{pt}\mathcal{L}_{pt} + \lambda_{conf}\mathcal{L}_{conf} + \lambda_{id}\mathcal{L}_{id}$
    Lpt, Lconf et Lid traduisent respectivement la mean-squared error sur les coordonnées, la Loss de confiance et la cross entropy pour la Loss de classification
    - ƛ_conf = 0.1 pour les cellules sans objets (améliore stabilité)
    - ƛ_conf = 5 sinon
    - ƛ_pt = ƛ_id = 1
  - Il est possible d'avoir plusieurs objets par cellule. Le programme autorise jusqu'à 5 prédictions par cellule, ce qui est suffisant dans les cas réels

Description du modèle

- **Prédiction:**
  - La prédiction est faite en n’appelant qu’une seule fois le réseau
  - Les prédictions avec une confiance trop faible sont supprimées via un seuil
  - Si un objet est détecté dans plusieurs cellules avec une grande confiance, les prédictions des 3x3 cellules voisines de la cellule avec la meilleure confiance sont moyennées avec une pondération selon la confiance
  - Le réseau fournit la projection 2D du centroid de l'objet et de de la boîte englobante. Nous estimons ensuite la pose 6D avec une méthode d’estimation de pose Perspective-n-Point PnP

Détails sur la mise en oeuvre

- **Initialisation :**
  - Problème :
    - Les estimations de pose lors des premières étapes sont inexactes et les valeurs de confiance peu fiable au départ
  - Solution :
    - Fixer le paramètre de régularisation à 0 pour la confiance au départ puis à 5 pour les cellules contenant un objet, 0.1 pour les autres cellules
    - Fixer la netteté de la fonction de confiance à 2 et du seuil de distance à 30 pixels

- **Optimisation :**
  - Descente du gradient stochastique

- **Sur-apprentissage :**
  - Augmentation extensive des données (HSV, mise à l'échelle, translation)

- **Implémentation effectuée sur PyTorch**

Description des expériences

- **2 contextes :**
  - Un seul objet 6D à estimer par image
  - Plusieurs objets 6D à estimer par image

- **Données utilisées :**
  - LineMod : 15783 images pour 13 objets
  - OCCLUSION : similaires à LineMod mais les objets sont occultés par l'encombrement
  - Même données que les modèles de l'état de l'art

- **Mesures effectuées**
  - Erreur de reprojection 2D
  - Score IoU Intersection over Union
  - Moyenne de la distance 3D des sommets du modèle

Détails sur les mesures

- **Erreur de reprojection 2D :**
  - Distance moyenne des sommets du maillage 3D de l'objet entre l'estimation et la vérité terrain
    - Choix : 5 pixels

- **Score IoU :**
  - Mesure les chevauchements entre la projection 3D du modèle et la vérité terrain
    - Choix : Chevauchement plus large que 0.5

- **Comparer la pose 6D :**
  - Distance moyenne entre les vrais coordonnées du sommets du maillage 3D et ceux de l'estimation
    - Choix : Inférieur à 10% du diamètre de l'objet

    - Cas particulier : les objets à rotation invariable (oeuf)

$$s = \frac{1}{|\mathcal{M}|} \sum_{x_1 \in \mathcal{M}} \min_{\mathcal{M}} \|(\mathbf{R}\mathbf{x} + \mathbf{t}) - (\hat{\mathbf{R}}\mathbf{x} + \hat{\mathbf{t}})\|$$

Estimation d’une pose d’un seul objet

- **Problème :**
  - Influence du background lors de l’apprentissage
- **Solution :**
  - Utilisation de masques de segmentations qui remplace le fond de manière aléatoire (Données : PASCAL VOC)

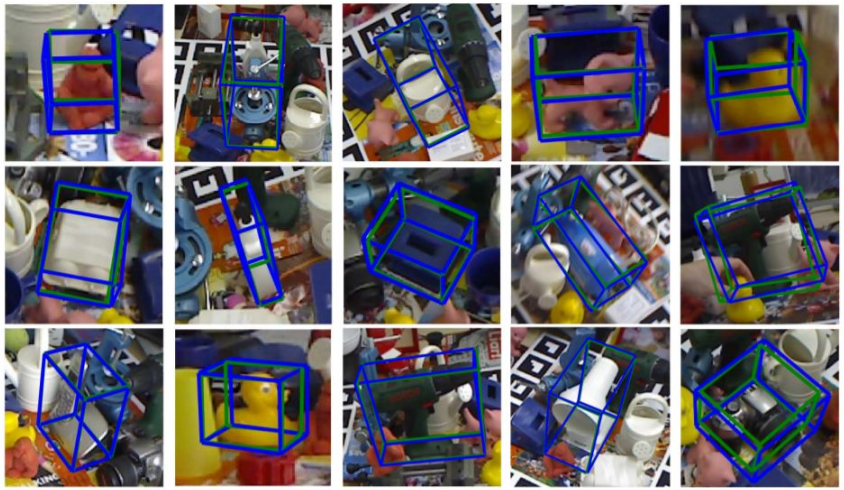

Figure 3: Pose estimation results of our approach. Note that our method can recover the 6D pose in these challenging scenarios, which involve significant amounts of clutter, occlusion and orientation ambiguity. In the last column, we show failure cases due to motion blur, severe occlusion and specularity (this figure is best viewed on a computer screen).

Estimation d'une pose d'un seul objet

Method	w/o Refinement			w/ Refinement	
Object	Brachmann [2]	BB8 [25]	OURS	Brachmann [2]	BB8 [25]
Ape	-	***95.3***	92.10	85.2	**96.6**
Benchvise	-	80.0	**95.06**	67.9	90.1
Cam	-	80.9	**93.24**	58.7	86.0
Can	-	84.1	**97.44**	70.8	91.2
Cat	-	97.0	**97.41**	84.2	98.8
Driller	-	74.1	***79.41***	73.9	**80.9**
Duck	-	81.2	**94.65**	73.1	92.2
Eggbox	-	87.9	**90.33**	83.1	91.0
Glue	-	89.0	**96.53**	74.2	92.3
Holepuncher	-	90.5	**92.86**	78.9	95.3
Iron	-	78.9	***82.94***	83.6	**84.8**
Lamp	-	74.4	**76.87**	64.0	75.8
Phone	-	77.6	**86.07**	60.6	85.3
Average	69.5	83.9	**90.37**	73.7	89.3

Table 1: Comparison of our approach with state-of-the-art algorithms on LINEMOD in terms of 2D reprojection error. We report percentages of correctly estimated poses. In

Estimation d'une pose d'un seul objet

Method	w/o Refinement		w/ Refinement
Object	SSD-6D [10]	OURS	SSD-6D [10]
Ape	98.46	**99.81**	99
Benchvise	**100**	99.90	**100**
Cam	99.53	**100**	99
Can	**100**	99.81	**100**
Cat	99.34	**99.90**	99
Duck	99.04	**100**	98
Glue	97.24	**99.81**	98
Holepuncher	98.95	**99.90**	99
Iron	99.65	**100**	99
Lamp	99.38	**100**	99
Phone	99.91	**100**	100
Average	99.22	**99.92**	99.4
Driller	-	**100**	99
Eggbox	-	**99.91**	99

Table 4: Comparison of our approach against [10] on LINEMOD using IoU metric. The authors of [10] were able to provide us the results of our approach w/o the refinement.

Estimation d'une pose d'un seul objet

Method	w/o Refinement				w/ Refinement		
Object	Brachmann [2]	BB8 [25]	SSD-6D [10]	OURS	Brachmann [2]	BB8 [25]	SSD-6D [10]
Ape	-	***27.9***	0	21.62	33.2	40.4	**65**
Benchvise	-	62.0	0.18	***81.80***	64.8	**91.8**	80
Cam	-	***40.1***	0.41	36.57	38.4	55.7	**78**
Can	-	48.1	1.35	***68.80***	62.9	64.1	**86**
Cat	-	***45.2***	0.51	41.82	42.7	62.6	**70**
Driller	-	58.6	2.58	***63.51***	61.9	**74.4**	73
Duck	-	***32.8***	0	27.23	30.2	44.3	**66**
Eggbox	-	40.0	8.9	***69.58***	49.9	57.8	**100**
Glue	-	27.0	0	***80.02***	31.2	41.2	**100**
Holepuncher	-	42.4	0.30	***42.63***	52.8	**67.2**	49
Iron	-	67.0	8.86	***74.97***	80.0	**84.7**	78
Lamp	-	39.9	8.20	***71.11***	67.0	**76.5**	73
Phone	-	35.2	0.18	***47.74***	38.1	54.0	**79**
Average	32.3	43.6	2.42	***55.95***	50.2	62.7	**79**

Table 2: Comparison of our approach with state-of-the-art algorithms on LINEMOD in terms of ADD metric. We report percentages of correctly estimated poses.

Estimation d'une pose d'un seul objet

Threshold	10%		30%		50%	
Object	[10]	OURS	[10]	OURS	[10]	OURS
Ape	0	**21.62**	5.62	**70.67**	19.95	**88.10**
Benchvise	0.18	**81.80**	2.07	**91.07**	10.62	**98.85**
Cam	0.41	**36.57**	34.52	**81.57**	63.54	**94.80**
Can	1.35	**68.80**	61.43	**99.02**	85.49	**99.90**
Cat	0.51	**41.82**	36.87	**90.62**	64.04	**98.80**
Driller	2.58	**63.51**	56.01	**99.01**	84.86	**99.80**
Duck	0	**27.23**	5.56	**70.70**	32.65	**89.39**
Eggbox	8.9	**69.58**	24.61	**81.31**	48.41	**98.31**
Glue	0	**80.02**	14.18	**89.00**	26.94	**97.20**
Holepuncher	0.30	**42.63**	18.23	**85.54**	38.75	**96.29**
Iron	8.86	**74.97**	59.26	**98.88**	88.31	**99.39**
Lamp	8.20	**71.11**	57.64	**98.85**	81.03	**99.62**
Phone	0.18	**47.74**	35.55	**91.07**	61.22	**98.85**
Average	2.42	**55.95**	31.65	**88.25**	54.29	**96.78**

Table 3: Comparison of our approach with SSD-6D [10] without refinement using different thresholds for the 6D pose metric.

Vitesse globale pour le cas d'une pose d'un seul objet

Method	Overall Speed	Refinement runtime
Brachmann et al. [2]	2 fps	100 ms/object
Rad & Lepetit [25]	3 fps	21 ms/object
Kehl et al. [10]	10 fps	24 ms/object
OURS	50 fps	-

Table 5: Comparison of the overall computational runtime of our approach in comparison to [2, 10, 25]. We further provide the computational runtime induced by the pose refinement stage of [2, 10, 25]

Estimation de pose de plusieurs objets

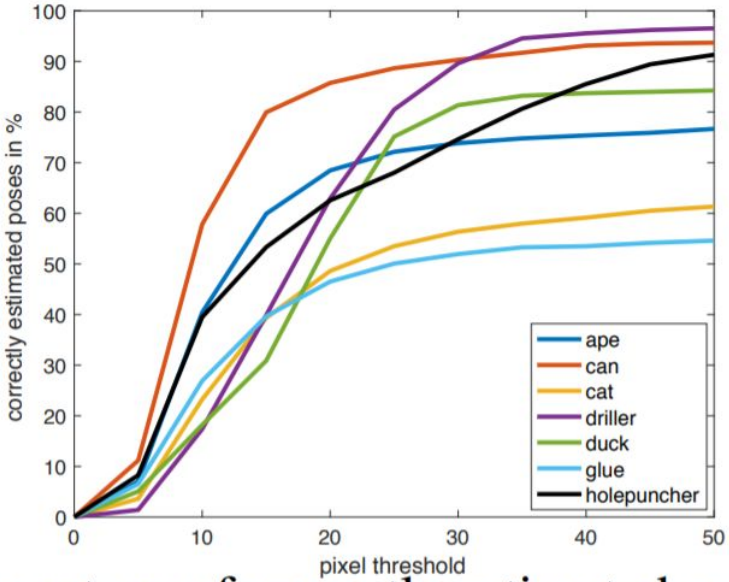

Figure 4: Percentage of correctly estimated poses as a function of the projection error for different objects of the Occlusion dataset [2].

Estimation de pose de plusieurs objets

Method	MAP
Hinterstoisser et al. [7]	0.21
Brachmann et al. [2]	0.51
Kehl et al. [10]	0.38
OURS	0.48

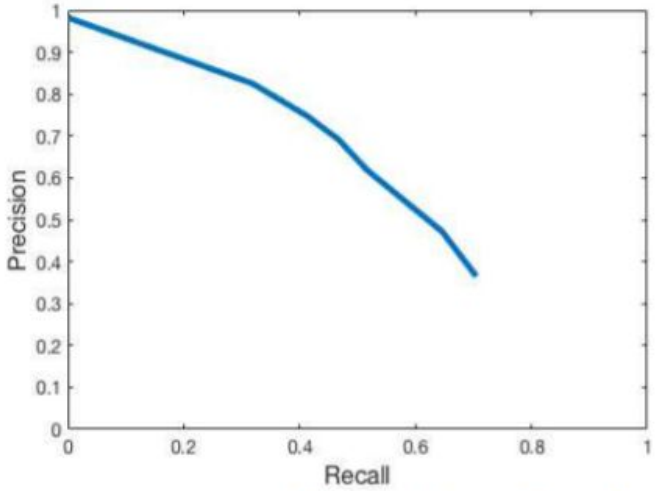

Table 6: The detection experiment on the Occlusion dataset [2]. (Left) Precision-recall plot. (Right)

Temps d'exécution

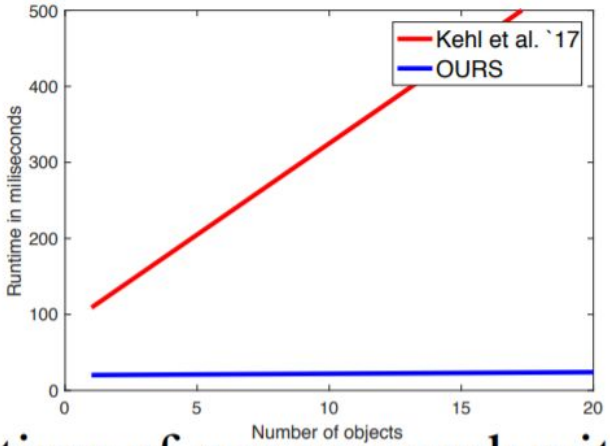

Figure 5: The runtime of our approach with increasing number of objects as compared to that of [10].

- Temps d’exécution : 0.2 ms/objet

Performance selon la résolution en entrée

Resolution	2D projection metric	Speed
416×416	89.71	94 fps
480×480	90.00	67 fps
544×544	90.37	50 fps
688×688	90.65	43 fps

Table 7: Accuracy/speed trade-off of our method on the LINEMOD dataset. Accuracy reported is the percentage of correctly estimated poses w.r.t the 2D projection error. The same network model is used for all four input resolutions. Timings are on a Titan X (Pascal) GPU.

Conclusion

- Nouvelle structure basée sur CNN pour l’estimation de la pose 6D
- Une seule prédiction aussi précise que ceux de l’état de l’art
- Pas besoin de raffinement
- Méthode en temps réel
- Meilleurs performances en terme de temps d’exécution