Robust vision for robotics: leveraging movement to improve perception skills

Kick off chaire certif AI

presented by **Joris Guérin**

on November 23 2020

Table of content

Outline

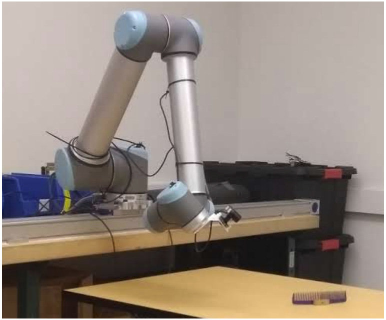
1. Introduction
2. Multi-view unsupervised robotic sorting
3. Semantically meaningful view selection
4. Robust object detection for periodic trajectories
5. Towards runtime monitoring

Today's talk

- Guérin, J., Thiery, S., Nyiri, E., & Gibaru, O. (2018). **Unsupervised robotic sorting: Towards autonomous decision making robots.** In International Journal of Artificial Intelligence & Applications (IJAIA), 2018.
- Guérin, J., Gibaru, O., Nyiri, E., Thieryl, S., & Boots, B. (2018, October). **Semantically Meaningful View Selection.** In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 1061-1066). IEEE.
- Guérin, J., Canuto, AMDP., & Goncalves, LMG. (2020). **Robust Detection of Objects under Periodic Motion with Gaussian Process Filtering.** to appear in International Conference on Machine Learning and Applications (ICMLA) 2020

Robust vision for robotics: leveraging movement to improve perception skills

On-board cameras



The camera pose can be controlled

Monitoring a robotic platform

Include the specificities of the robot movements in the vision algorithms

Outline

1. Introduction

2. Multi-view unsupervised robotic sorting

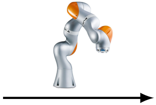
3. Semantically meaningful view selection

4. Robust object detection for periodic trajectories

5. Towards runtime monitoring

Unsupervised Robotic Sorting

Robotic sorting



Improve flexibility

Baseline approach

URS pipeline

Physical objects → **Image acquisition** → Digital images → **Image Clustering** → Classes

Image clustering pipeline

Images → Pretrained Xception Feature Extractor → Features → Agglomerative Clustering → Classes

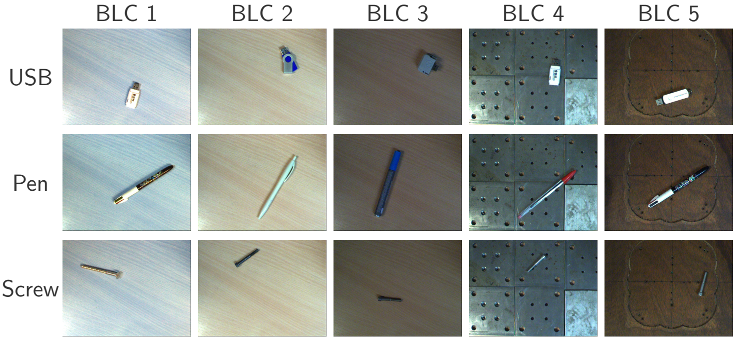
Demonstration

Robustness testing dataset

Dataset statistics

#Images	Images size	#Classes	#Images per class	#BLC
560	640 x 480	7	12 to 24	5

Example images



Robustness testing dataset

Dataset statistics

#Images	Images size	#Classes	#Images per class	#BLC
560	640 x 480	7	12 to 24	5

Artificially modified brightness

Results

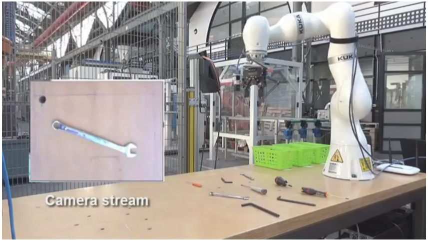
Clustering results on the tool clustering dataset

	BLC1	BLC2	BLC3	BLC4	BLC5
NMI	0.86	0.90	0.84	0.69	0.83
Purity	*0.85*	*0.90*	*0.85*	*0.69*	*0.81*

Clustering results for artificially modified lighting conditions

	Very dark	Dark	Normal	Bright	Very bright
NMI	0.77	0.88	0.90	0.84	0.73
Purity	*0.77*	*0.89*	*0.90*	*0.84*	*0.74*

Possible improvement strategies



- Top-down perpendicular views
- Xception + Agglomerative

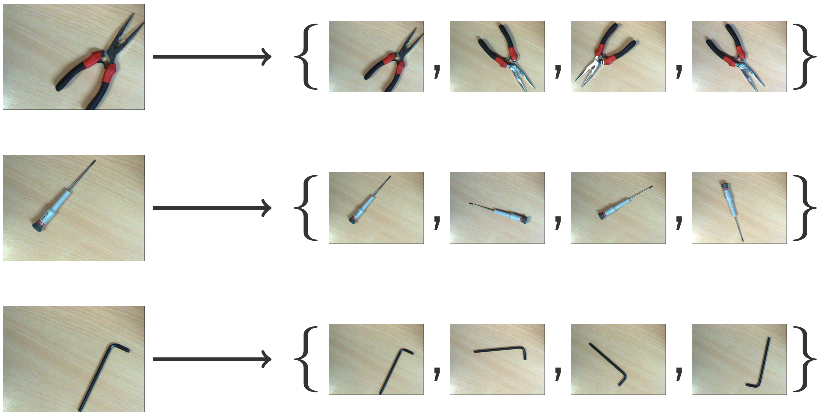
Lighting conditions
Scene layout
Objects poses
↓
Physical Objects → [Image Acquisition] → Image dataset → [*Image Clustering*] → Classes
↑
Camera poses
Camera parameters

Unmastered task variations
Tunable action levers

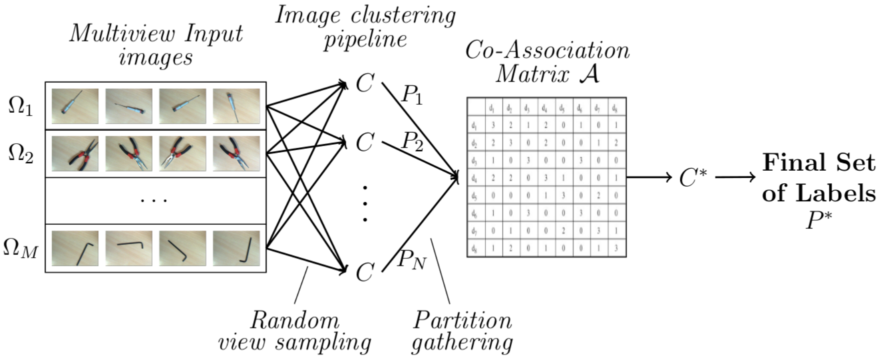
From Single-View to Multi-View Image Clustering

Single-view input

Multi-view input



Proposed ensemble clustering approach



Where

$$C \text{ and } C^* \text{ are respectively } \{\text{Xception} + \text{AC}\} \text{ and AC},$$

$$\mathcal{A}_{pq} = \frac{1}{N}\sum_{t=1}^{N} \delta(P_t(\Omega_p), P_t(\Omega_q)),$$

$$\delta(.,.) \text{ is the Kronecker symbol.}$$

Results

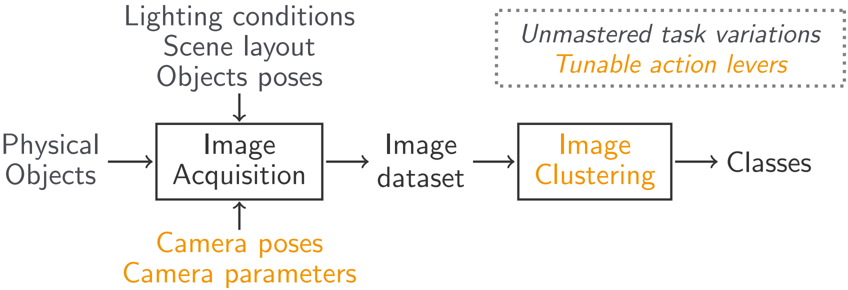
Clustering results on the tool clustering dataset

		BLC1	BLC2	BLC3	BLC4	BLC5
Single-view	NMI	0.86	0.90	0.84	0.69	0.83
	Purity	0.85	0.90	0.85	0.69	0.81
Multi-view	NMI	**0.95**	**1.00**	**0.95**	**0.84**	**0.95**
	Purity	**0.96**	**1.00**	**0.96**	**0.82**	**0.96**

Clustering results for artificially modified lighting conditions

		Very dark	Dark	Normal	Bright	Very bright
Single-view	NMI	0.77	0.88	0.90	0.84	0.73
	Purity	0.77	0.89	0.90	0.84	0.74
Multi-view	NMI	**0.91**	**1.00**	**1.00**	**0.96**	**0.84**
	Purity	**0.93**	**1.00**	**1.00**	**0.96**	**0.86**

Conclusions



Average result improvement:

NMI: $0.816 \rightarrow 0.939$

Purity: $0.816 \rightarrow 0.933$

Outline

1. Introduction

2. Multi-view unsupervised robotic sorting

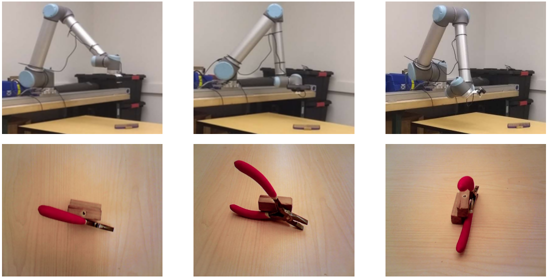
3. Semantically meaningful view selection

4. Robust object detection for periodic trajectories

5. Towards runtime monitoring

View selection problem

Importance of view selection:



Top view   Good view   Bad view

Objective:

$\longrightarrow \boxed{\text{Model}} \longrightarrow p_{c^*}$

Top perpendicular view   Optimal camera pose

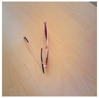
Building a large multi-view dataset

Example: 1 object in 1 pose

Top view

$(45°, 45°)$

$(45°, 225°)$



$(60°, 45°)$

$(60°, 225°)$

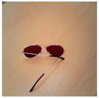
$(75°, 45°)$

$(75°, 225°)$

$(45°, 135°)$

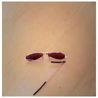
$(45°, 315°)$

$(60°, 135°)$



$(60°, 315°)$

$(75°, 135°)$



$(75°, 315°)$

Views are parameterized by two angles θ and φ

Dataset statistics

# Classes	# Object/class (*total*)	# Poses/object (*total*)	# Views/pose (*total*)
29	4-6 (*144*)	3 (*432*)	17-22 (*9112*)

Fitting a "Clusterability score" to the images

Estimating the quality of an image for clustering

- Sample N clustering problem (3×10^7)
- For each clustering problem cp:
  - Compute the individual Fowlkes-Mallows index of each image:

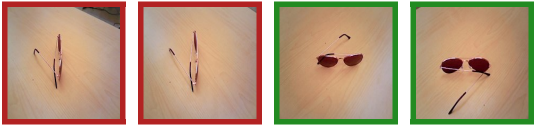
$$FMI^{i}_{cp} = \frac{TP_i}{\sqrt{(TP_i+FP_i)(TP_i+FN_i)}}$$

- Compute the Monte Carlo estimate of the clusterability index:

$$S(I) = \sum_{\text{cp}} FMI^{I}_{\text{cp}} / N^{I}_{\text{cp}}$$

N^{I}_{cp}, number of cp in which I is present

Qualitative validation



Reformulation of Semantic View Selection

Initial regression formultaion

$\longrightarrow \boxed{\text{Model}} \longrightarrow p_{c^*}$

Top perpendicular view

Optimal camera pose

Reformatting as a classification problem

Top perpendicular view

P_{cam}

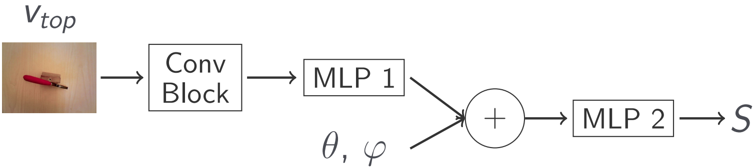
Proposed camera pose

$\longrightarrow \boxed{\text{Model}} \longrightarrow S(I)$

Semantic score

Training process

Network architecture



Data splitting

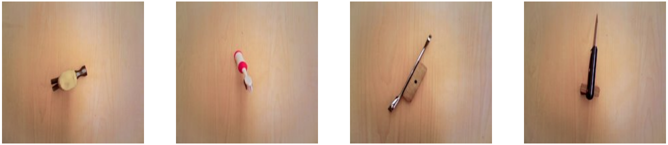
Clusterability index fitting	24 classes	
Neural network parameter selection	Training: 19	Testing: 5
Semantic View Predictor validation	5 classes	

Results

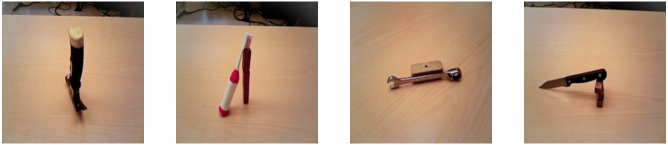
Quantitative results

		FM	NMI	PUR
	TOP	0.44	0.51	0.70
XCE_AGG	RAND	0.48	0.56	0.74
	SV-net	**0.55**	**0.63**	**0.78**

Qualitative evaluation



Example top views



Associated SV-net selections

Outline

1. Introduction

2. Multi-view unsupervised robotic sorting

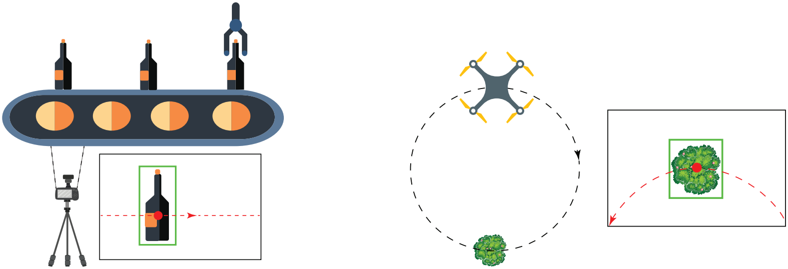
3. Semantically meaningful view selection

4. Robust object detection for periodic trajectories

5. Towards runtime monitoring

Problem definition

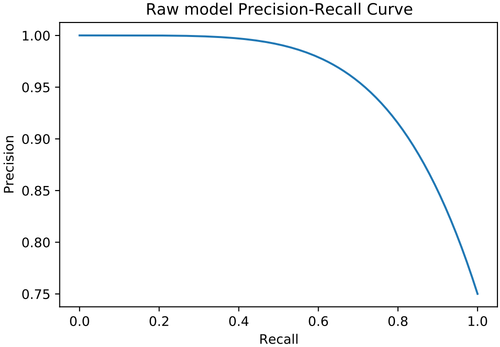
Object with a periodic trajectory



Imperfect Object Detection model

Model characterization

Precision-recall curve

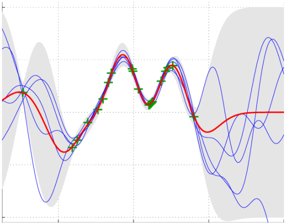


Parametric curve parameterized by the threshold on model scores for accepting predicted bounding boxes

Objective: Use the pattern in the object trajectories to filter the predictions and improve this curve

Big picture

Fit a Gaussian Process model to the trajectories

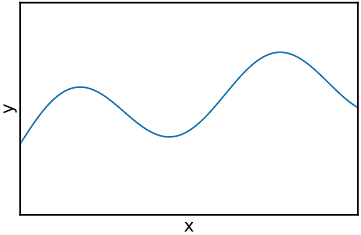


$$t_{pred} = \mu_{GP}(X, Y) + \sigma_{GP}(X, Y)$$

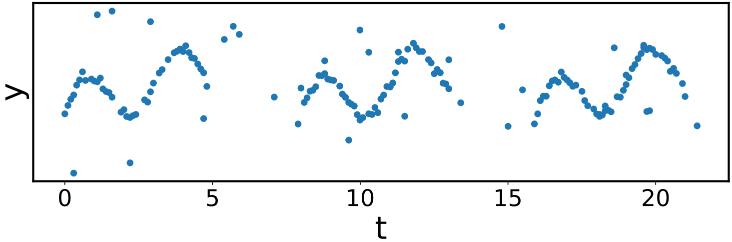
Use it at inference time to filter out wrong boxes

$$\sigma_{GP}(X, Y) < \max(\sigma_{training})$$
$$t_{pred} - t < \max(\sigma_{training})$$

First step: Collect training data

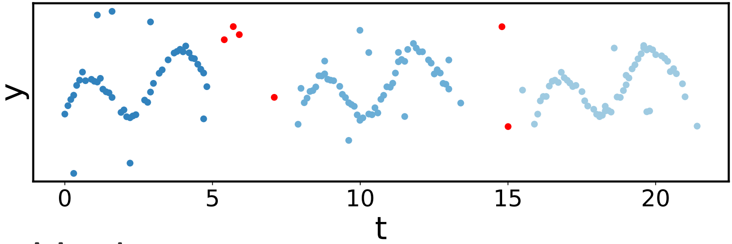


Run the OD model with high recall to gather training data

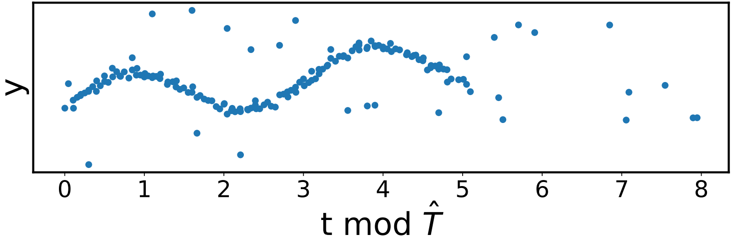


Step 2: Align trajectories

Density based clustering along time to identify object cycle

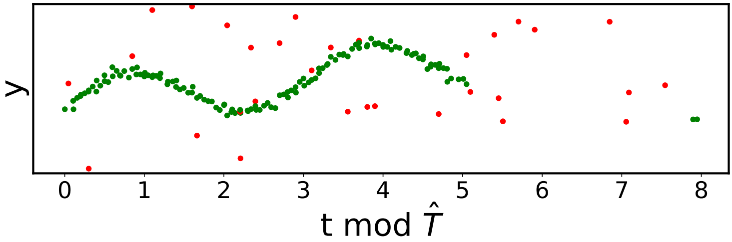


Align training data

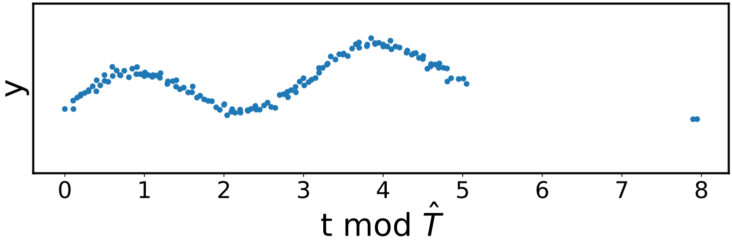


Step 3: Remove outliers

Density based clustering on $\{$X, Y, $t_{new}\}$ data

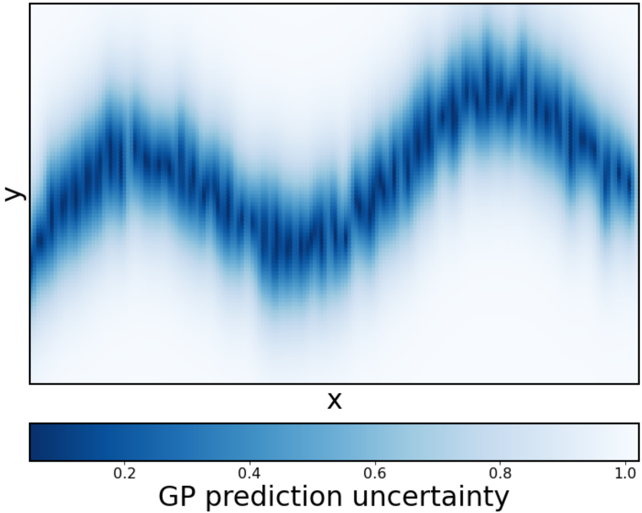


Clean training data for Gaussian Process

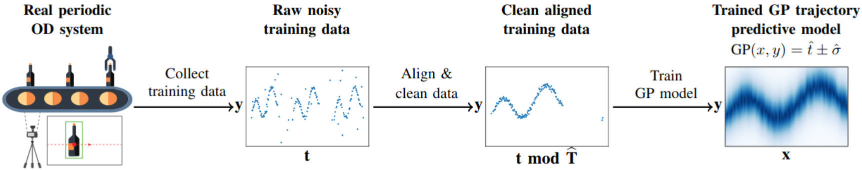


Step 4: Fit Gaussian Process model

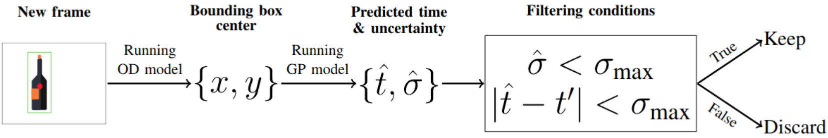
Visualize GP uncertainty in image plane



Full pipeline recap



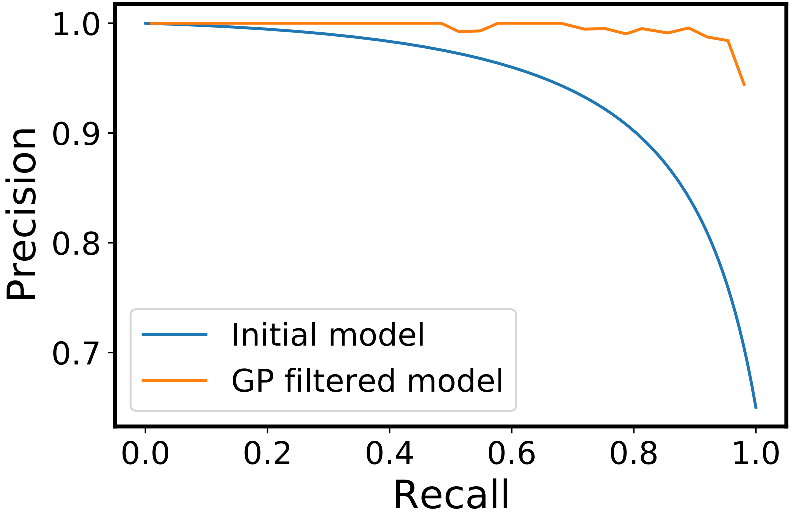
(a) Training



(b) Inference

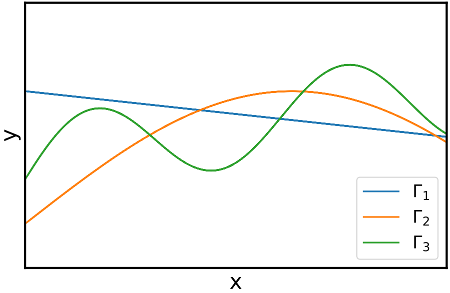
Results

New Precision-Recall curve

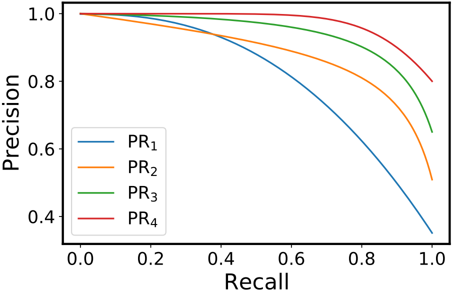


More complete experiments

Different trajectories



Different models



Results

Average Precision (AP)

OD model		PR_1	PR_2	PR_3	PR_4
Reference		0.811	0.883	0.942	0.975
Auto	Γ_1	0.483	0.606	**0.959**	**0.985**
clean-	Γ_2	0.403	0.523	**0.960**	0.974
ing	Γ_3	0.402	0.523	**0.981**	**0.986**
Manual	Γ_1	**0.940**	**0.966**	**0.959**	**0.986**
clean-	Γ_2	**0.979**	**0.978**	**0.988**	**0.980**
ing	Γ_3	**0.977**	**0.966**	**0.974**	**0.978**

Results

Optimal F1-score (oF1)

OD model		PR_1	PR_2	PR_3	PR_4
Reference		0.714	0.811	0.865	0.902
Auto	Γ_1	0.542	0.749	**0.964**	**0.994**
clean-	Γ_2	0.521	0.624	**0.971**	**0.992**
ing	Γ_3	0.496	0.697	**0.987**	**0.998**
Manual	Γ_1	**0.943**	**0.983**	**0.973**	**0.998**
clean-	Γ_2	**0.983**	**0.990**	**0.994**	**0.983**
ing	Γ_3	**0.977**	**0.986**	**0.990**	**0.994**

Outline

1. Introduction

2. Multi-view unsupervised robotic sorting

3. Semantically meaningful view selection

4. Robust object detection for periodic trajectories

5. **Towards runtime monitoring**

Take home messages from previous work

- Good computer vision algorithms are crucial for successful implementations of robotic tasks requiring visual perception
- However, the control and mechanics components of the problems should not be decorrelated from the vision modules
- Including knowledge about the dynamic of the problem and including the control loop in the vision modules can enable to improve the applications more than any improvement on CV algorithms alone.

Next: My vision of "Runtime Verification for Critical ML Applications"

Specificities of applications containing ML components

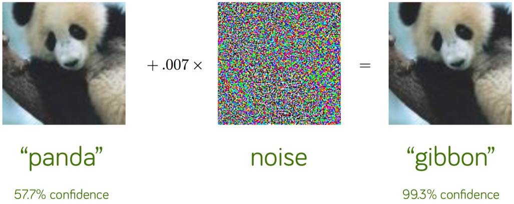
- Has enabled to solve more complex perception problems, where the **environments are unmastered and cannot be modeled accurately**. Safety guarantees are hard to obtain.
- Deal with **complex data structures** (e.g. Images) where logic rule based reasoning is hard to implement. State is hard to measure.
- Tend to be **overconfident**

Prediction: dog
Probability: 0.98

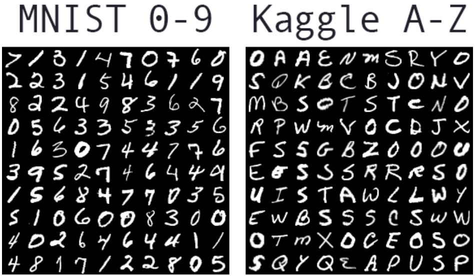
Prediction: dog
Probability: 0.95

Types of failures

- Malicious: Adversarial attacks



- Non malicious: Out of distribution



Types of monitoring

What do we know about the model?

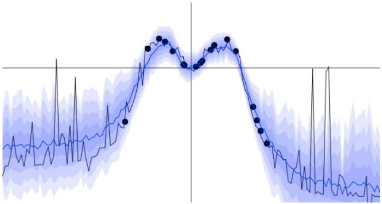
- White box model with access to training data (1),
- White box model without access to training data (2),
- Black box model (3).

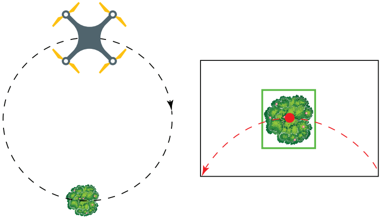
What do we monitor?

- Inputs entering the model (1),
- Model activations (1, 2),
- Model outputs (1, 2, 3),
- Info from outside the model (other sensors) (1, 2, 3).

Types of protection

- Evaluate uncertainty in model predictions well
- Detect potentially dangerous states of the system
- Find some patterns about expected outputs/behavior of the system and use it to ensure correctness





Interesting use cases?

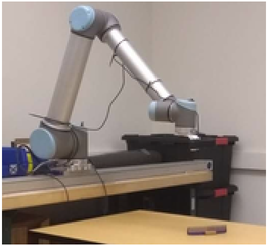
Robust vision for robotics: leveraging movement to improve perception skills

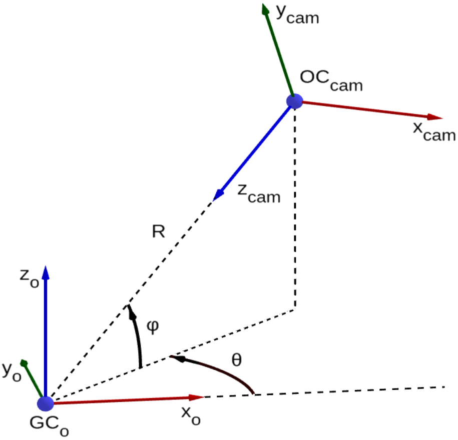
Kick off chaire certif AI

presented by **Joris Guérin**

on November 23 2020

View parameterization





Procedure:

- 3D camera
- Bounding box
- 75% of the image
- Parameterization