DaVinci Surgical Instruments Detection for Dexterity Tests

using Single Shot MultiBox Detector

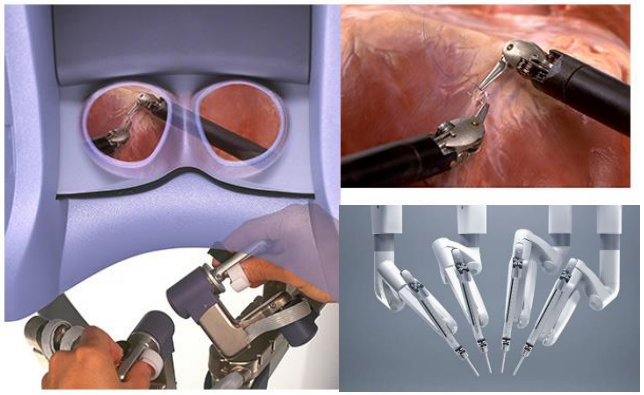
2016. 09. 19

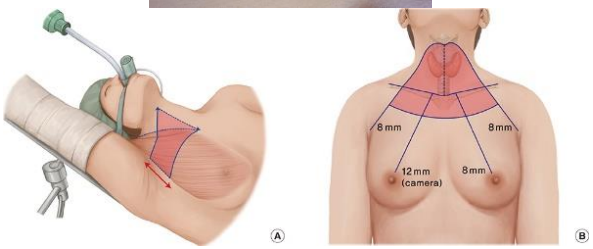
(협) 바이오엔지니어링

이동헌

Introduction

❖ Robotic Thyroidectomy

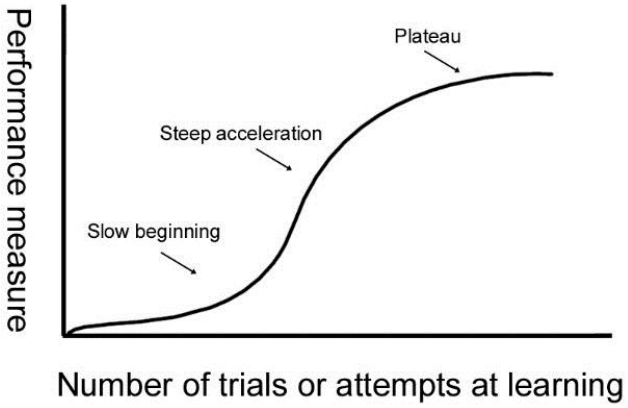




<Davinci Simulator>

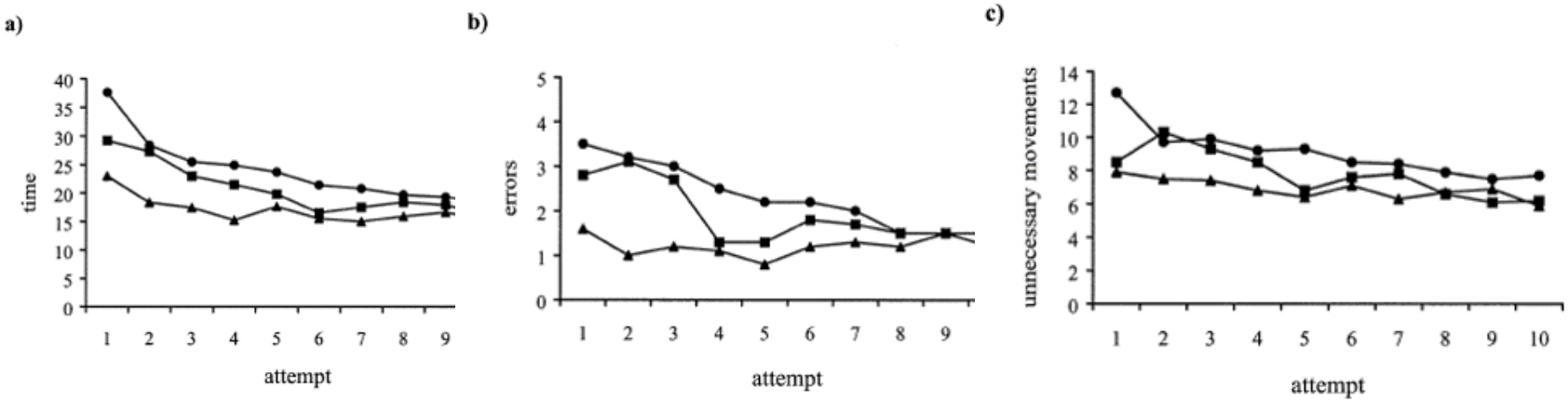
Learning Curve

- **수술의 숙련도**, 즉, 얼마나 정확하고 안전하게 수술을 시행할 수 있느냐를 평가하기 위한 개념으로 학습곡선(Learning Curve)이라는 개념이 도입됨
- 동일한 수술을 시행할 경우, 수술 후 합병증이 어느 정도의 확률로 발생하며, 수술 시간은 얼마나 걸리는 지를 측정하여 수술 숙련도를 평가함



- **시간, 수술도구의 움직임**과 주로 관련

e.g 수술도구의 총 움직인 거리, 수술도구가 화면 밖으로 나간 횟수, 수술도구간의 충돌 횟수 등

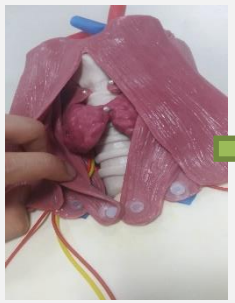


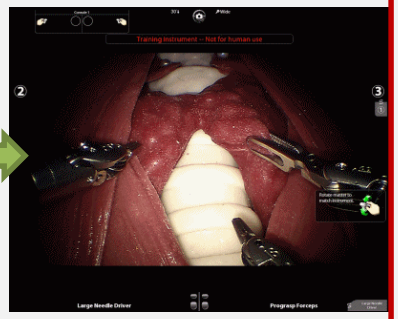
<Learning curves for masters (**filled triangles**), intermediates (**filled squares**), and beginners (**filled circles**) [1]>

연구 개요

내분비 외과

<훈련용 갑상선 모델 및 Simulator>





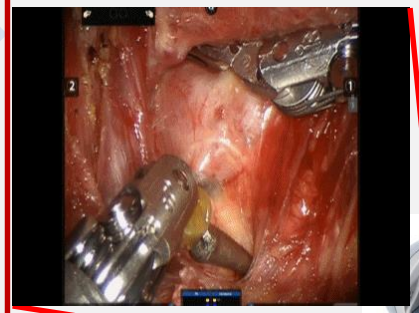
A (Expert), B (Trainee), C (Novice)
연습 횟수와 점수의 연관성

(≒ 운전면허시험)

Simulator 점수와
실제수술 지표 변화의
관련성 평가

의공학과

<알고리즘 개발>



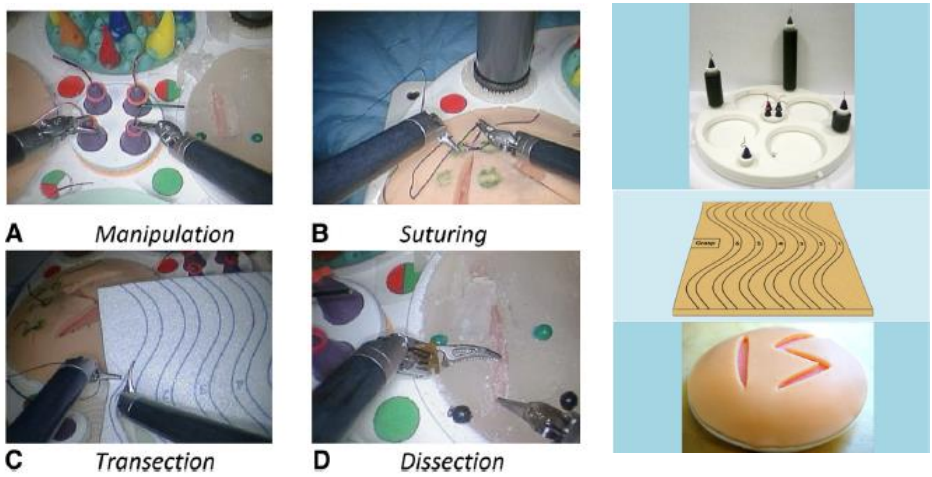
(≒ 실제 도로운전)

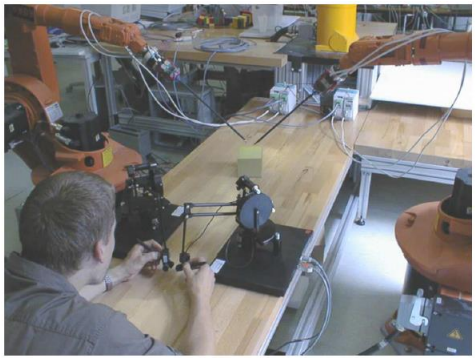
로봇 수술 숙련도 평가 모델과
트레이닝 시스템의 개발 및 적용

선행 연구 (In vitro)

- Manipulation, Suturing, Transection, Dissection 등 다양한 숙련도 평가 방법들이 존재함[2-3]

- In-vitro 환경의 숙련도 평가 방법이 아닌, 실제 수술 환경에서 숙련도 평가 방법 개발이 필요함





Kinematics[4]

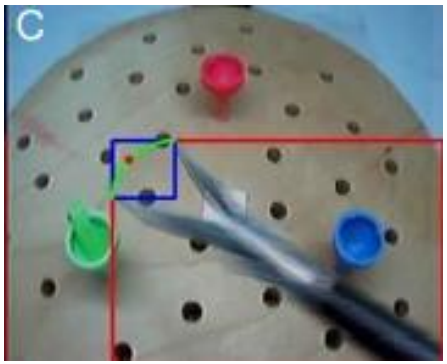
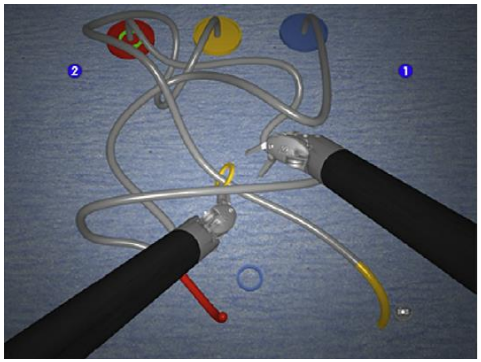
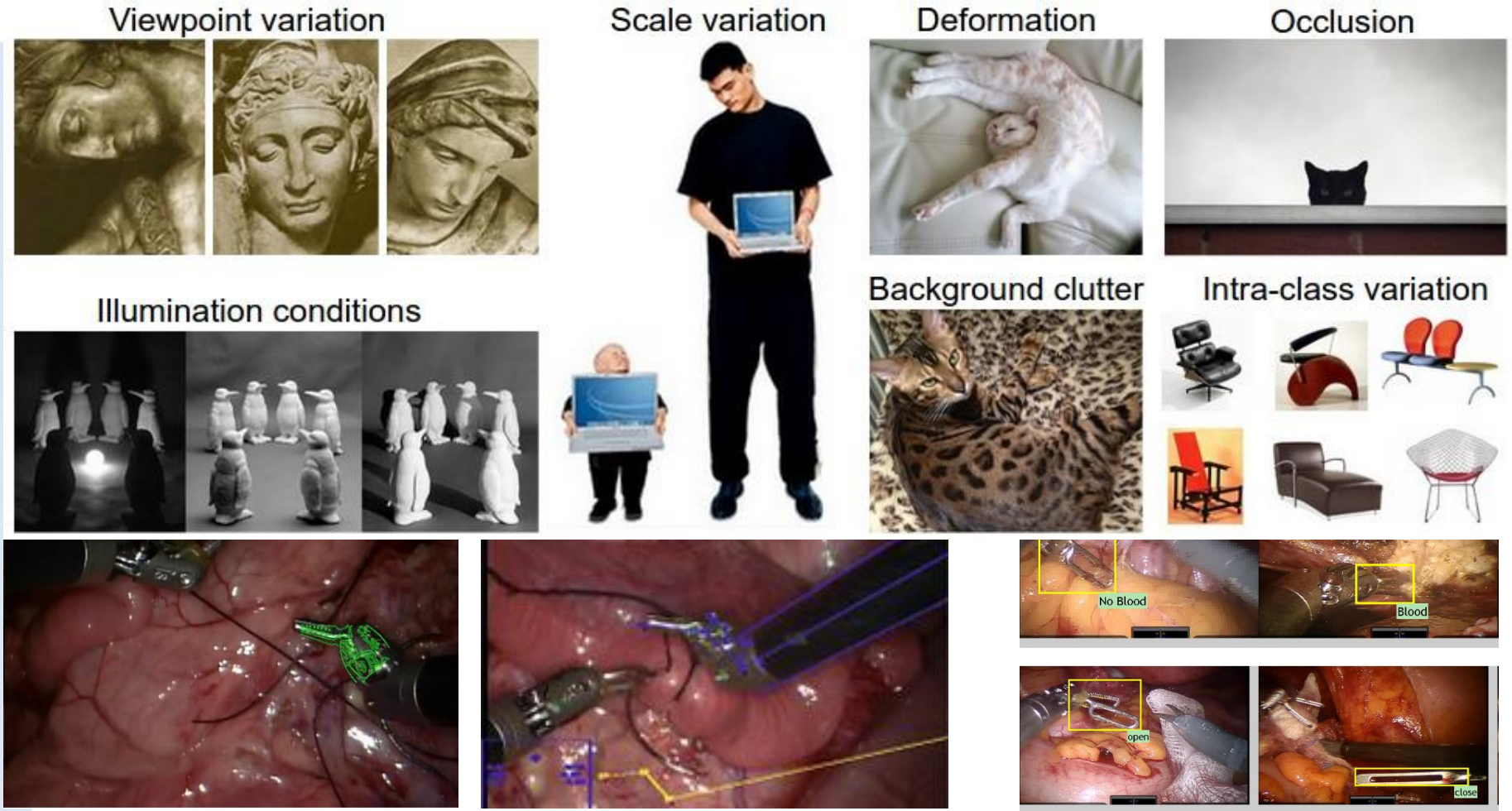


Image Processing[5]



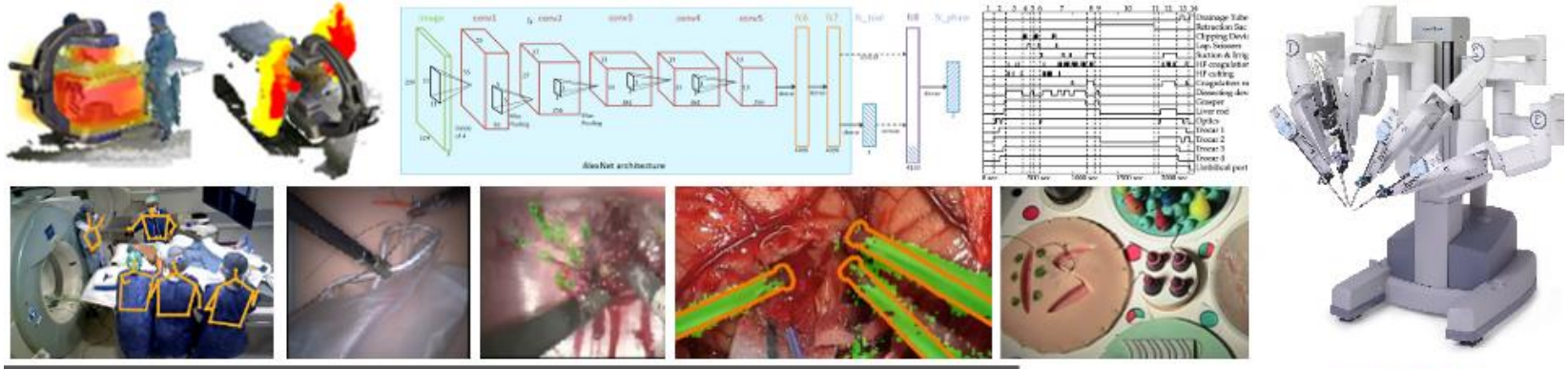
Virtual Kit[6]

선행 연구 (In vivo)



<Surgical Instrument Tracking (Animal)[7-8]>

<Surgical Instrument Semantic Attributes[9]>



M2CAI | 21 Oct Athens

MICCAI 2016 Athens GREECE

(1) Surgical workflow challenge

Frame	Phase
0	TrocarPlacement
1	TrocarPlacement
2	TrocarPlacement
...	
49	Preparation
...	

Grasper Hook Clipper

Bipolar Irrigator Scissors

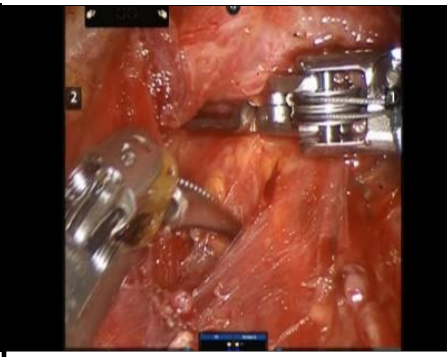
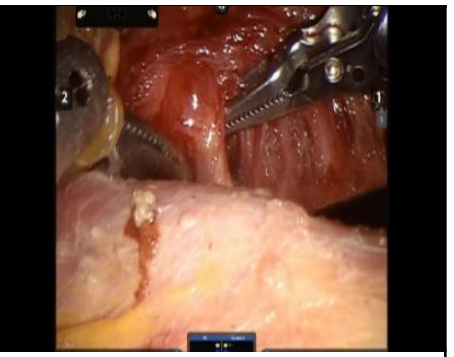
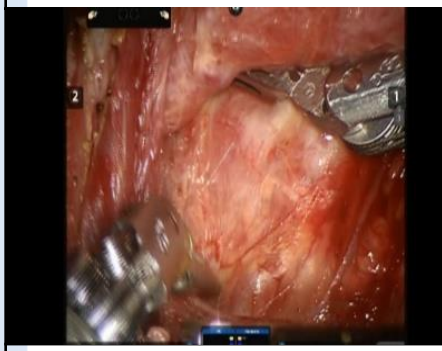
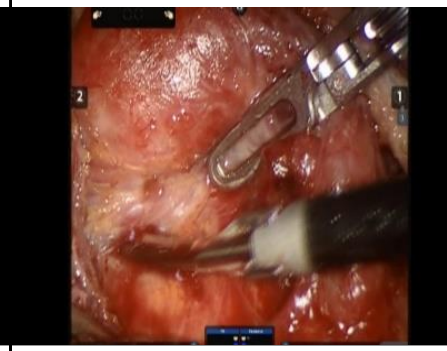
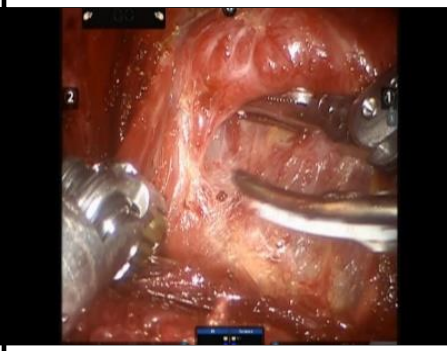
(2) Surgical tool detection challenge

Frame	Grasper	Bipolar	Hook	Scissors	Clipper	Irrigator	SpecimenBag
0	-1.5348	-3.0689	2.8234	-0.9753	-3.3533	-2.8772	-1.2033
25	2.8949	1.0443	0.38031	-3.6901	4.5647	-0.74929	-3.1698
50	-2.6808	1.3377	0.7014	-2.6383	0.20482	-4.2711	0.21033
75	-1.6789	0.20425	1.3347	1.3916	2.6992	2.7968	-3.2456

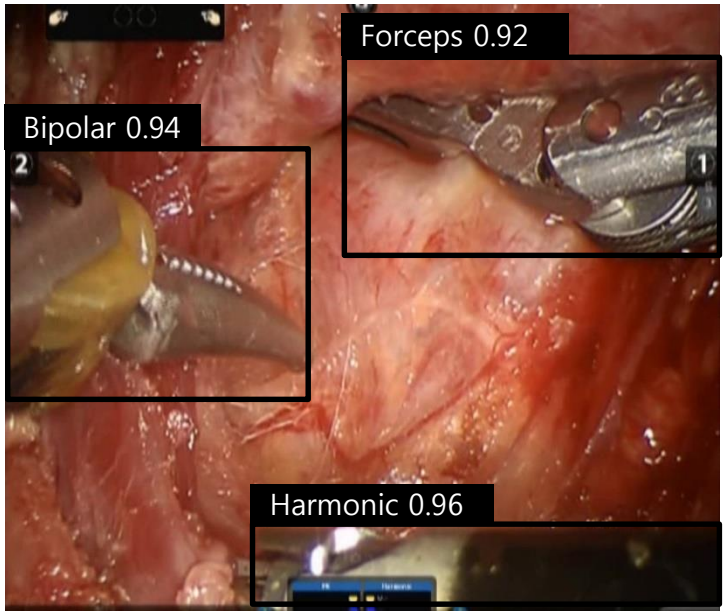
Specimen bag

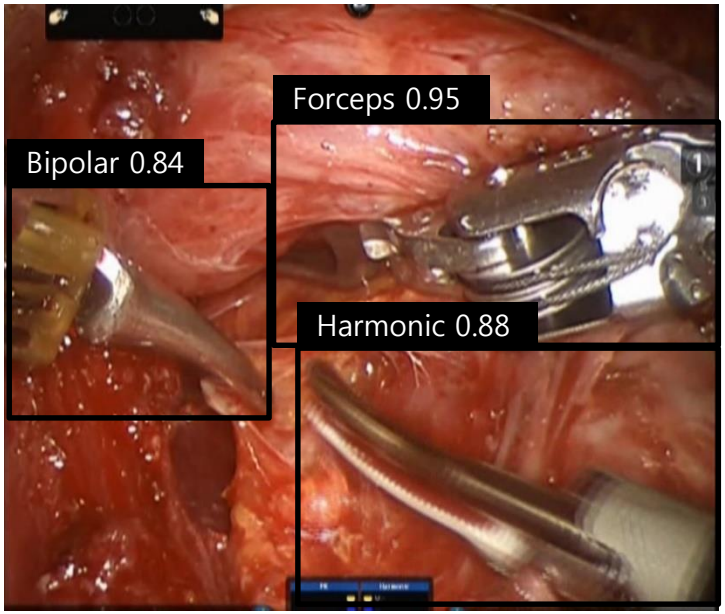
Methods

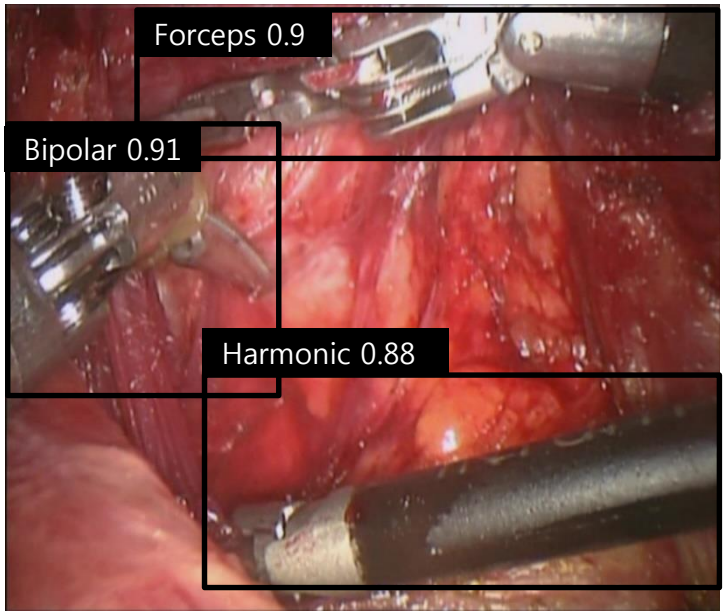
- 서울대학교병원 내분비외과 갑상선 수술 동영상(50례)
- 동영상의 전체 frame 개수: 292,880장 → 12,708장 (1/23 frames)

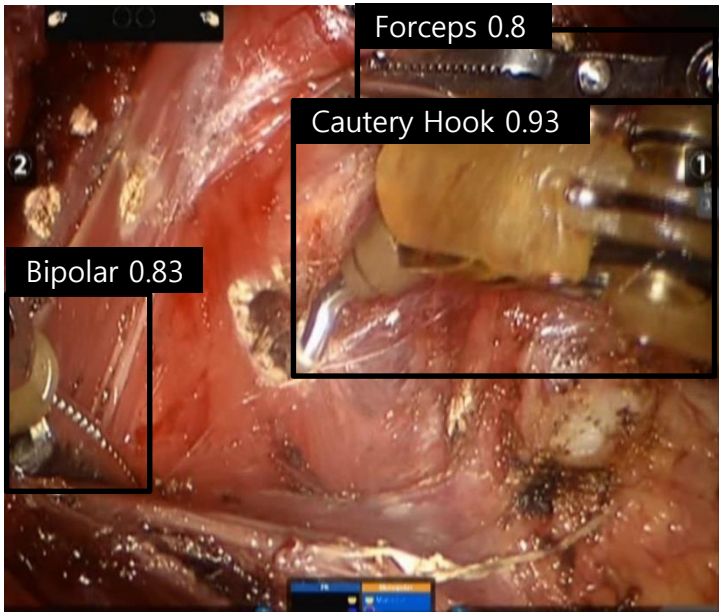
			
(7) 02:41(3,859 frames)	**(13)** 01:29(2,142 frames)	**(15)** 01:32(2,198 frames)	**(22)** 13:51(19,932 frames)
			
(24) 01:43(2,479 frames)	**(25)** 00:42(2,034 frames)	**(44)** 05:05(7,307 frames)	**(50)** 00:25(613 frames)

Expected Results



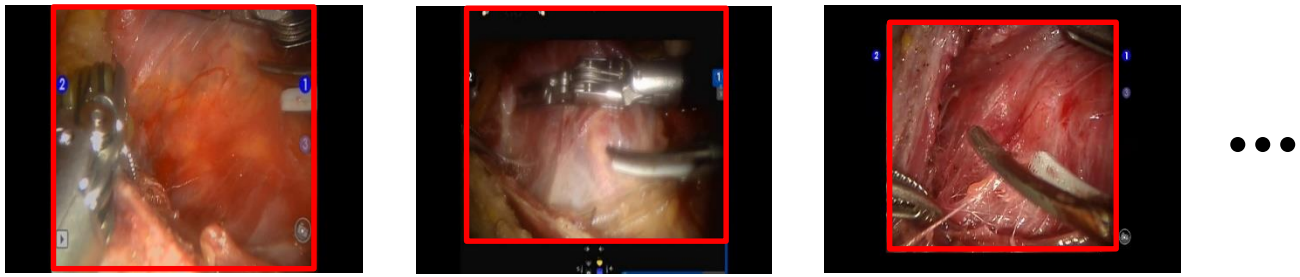






Preparing Dataset

① ROI 설정



② Resize

: 1440 x 1080 → 500 x 4xx 로 통일 (Reference dataset: VOC PASCAL)

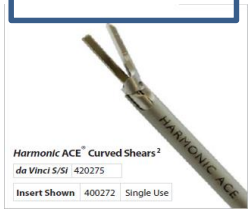
(※ 숙련도 **상, 중, 하** 고려)

Preparing Dataset

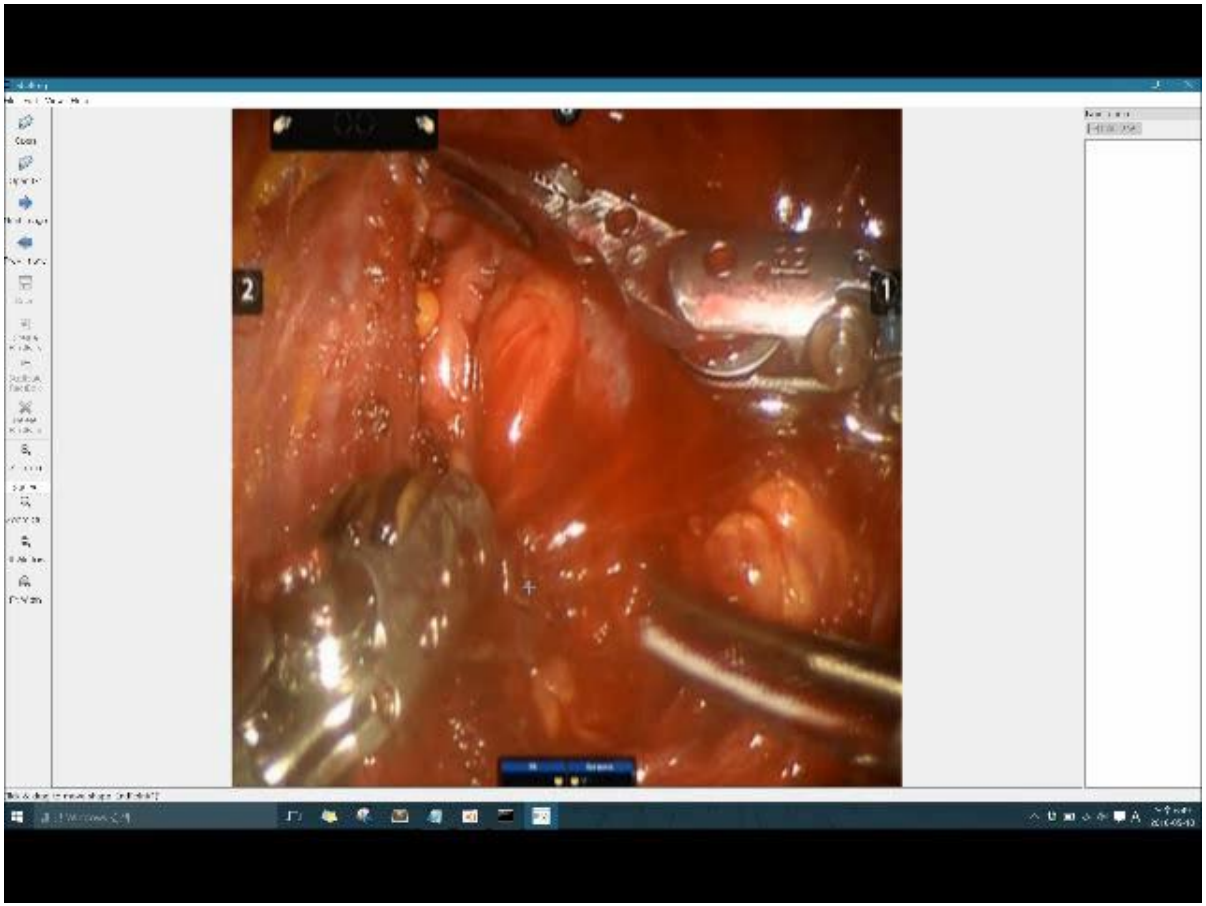
Bipolar

Forceps

Harmonic



Cautery Hook



Detection Algorithms(1)

1. Faster R-CNN (2015.06.04)

- **내용**: RPN(Region Proposal Network)를 제안. RCNN과 ConvNet을 공유하여 (Fast RCNN 대비) 계산시간 줄임
- **GitHub**: https://github.com/rbgirshick/py-faster-rcnn (Caffe)
  https://github.com/smallcorgi/Faster-RCNN_TF (Tensorflow)

2. SSD: Single Shot MultiBox Detector (2015.12.08)

- **내용**: Proposal 생성단계를 생략하여 detection 속도 향상
- **GitHub**: https://github.com/weiliu89/caffe/tree/ssd (Caffe)

3. ION: Inside Outside Network (2015.12.14)

- **내용**: (ROI 기준으로) Inside (Skip pooling을 multi-scale에 적용) + Outside (context 정보를 위하여 IRNN 적용)
- **GitHub**: -

Detection Algorithms(2)

4. MR-CNN (2015.05.07)

- **내용**: 여러 모양의 영역을 정의하여 CNN 입력으로 사용
- **GitHub**: https://github.com/gidariss/mrcnn-object-detection (Caffe)

5. A MultiPath Network for Object Detection (2016.04.07)

- **내용**: -
- **GitHub**: https://github.com/facebookresearch/multipathnet (Torch)

6. R-FCN: Region-based Fully Convolutional Networks

(2016.05.20)

- **내용**: Position sensitive score map을 제안하여 translation variance/invariance 특성을 모두 활용
- **GitHub**: https://github.com/daijifeng001/R-FCN (Caffe)

SSD: Single Shot MultiBox Detector

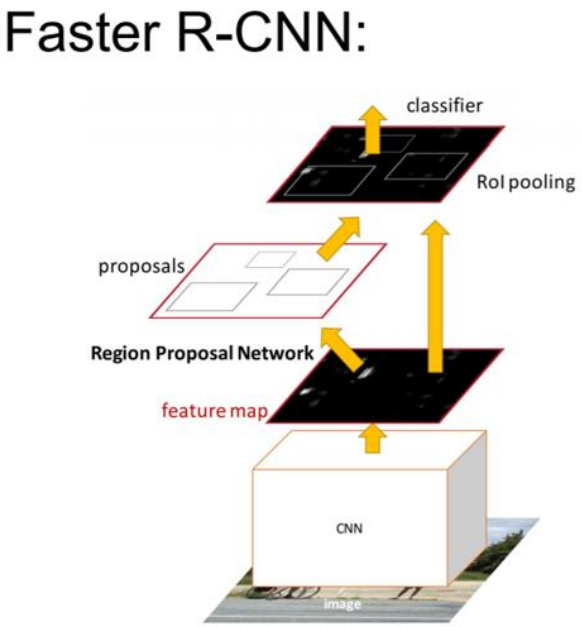
Abstract

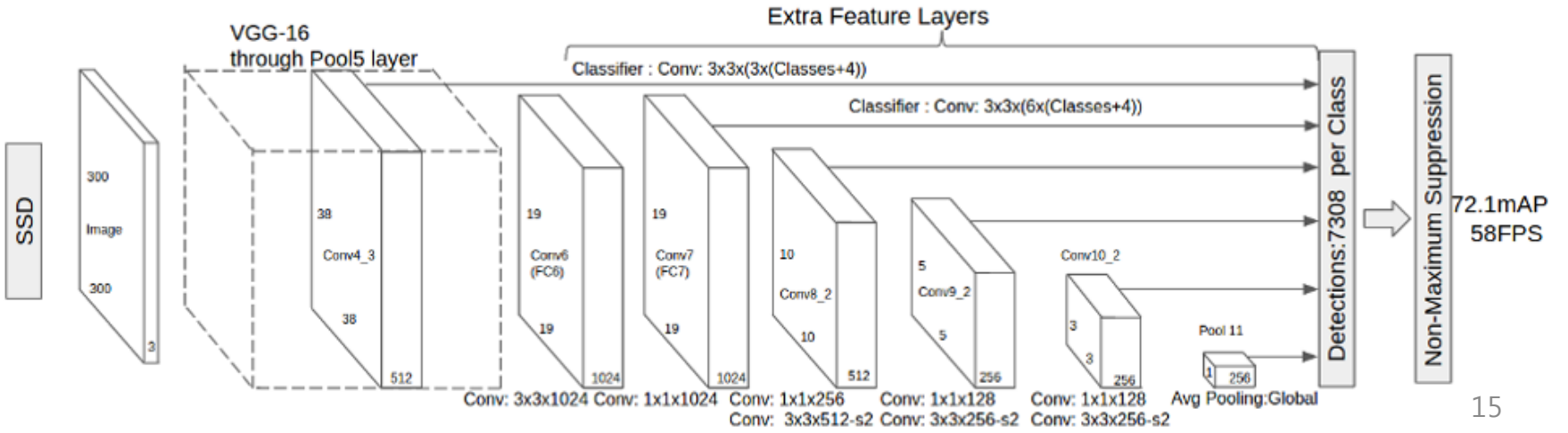
- **내용**
  - Region Proposal 부분을 제거하고 single network로 구성함
  - 여러 크기의 feature map에서 정해진 aspect ratio(가로/세로 비율)와 scale(크기)를 갖는 default box들로 나누어 인식을 수행

- **결과**
  - Region Proposal 방법과 비슷한 성능, 빠른 속도
  - 단일 network 제시 → training과 inference 같음

The Single Shot Detector (vs Faster R-CNN)

- **Region Proposal network 가 없는 구조**
  - → 정확하지만 계산량이 많음(embedded 어려움)
  - → 속도의 한계(실시간 x)

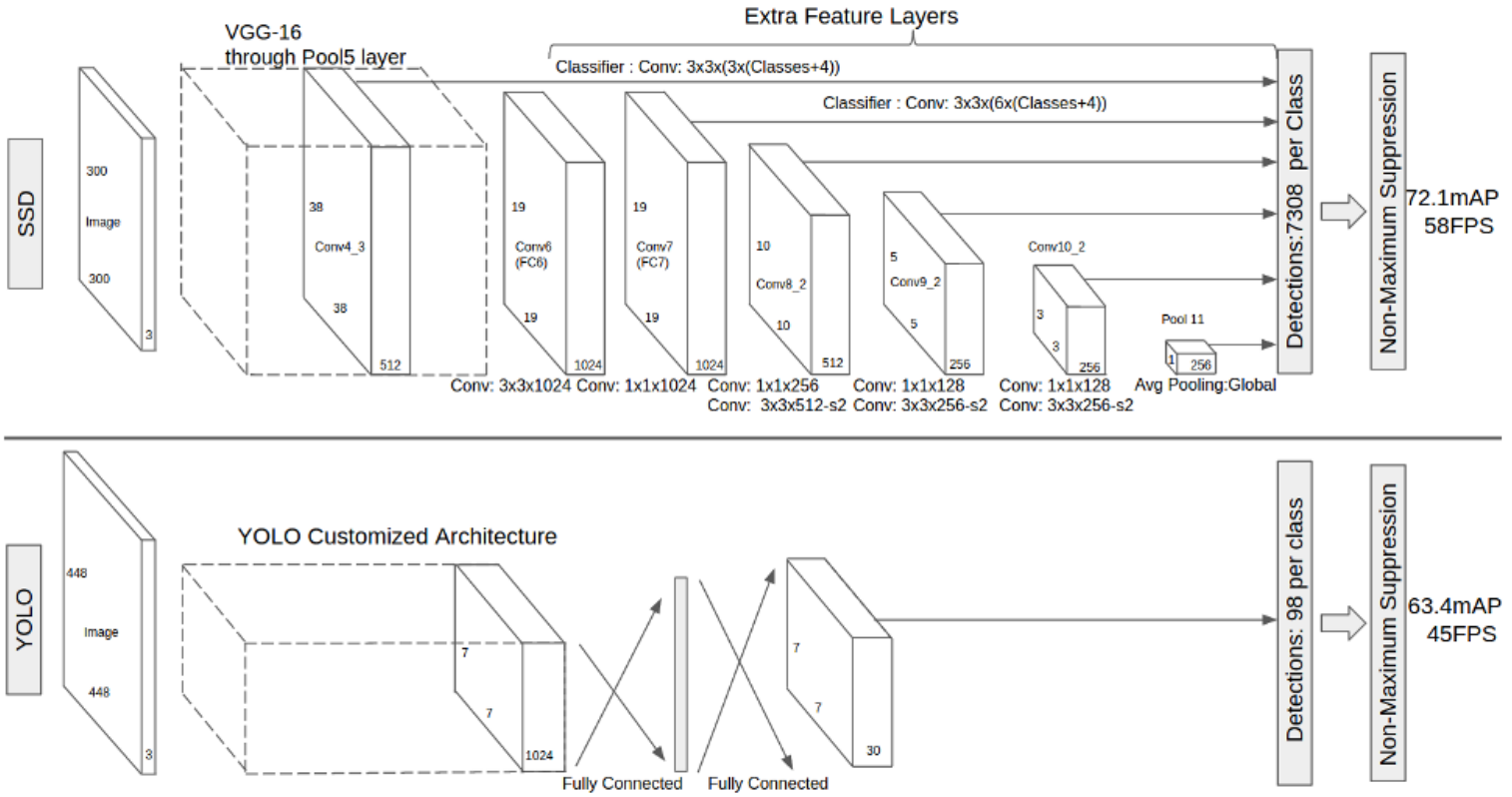




The Single Shot Detector (vs YOLO)

- **여러 크기의 feature map 사용:** scale 고려
- **작은 convolutional filter 사용:** aspect ratio 고려

  → YOLO와 비교했을 때, input size가 더 작아도 인식률이 높음



2.1 Model

(1) Base Network

: VGG(약간 변형해서 사용), ResNet 등

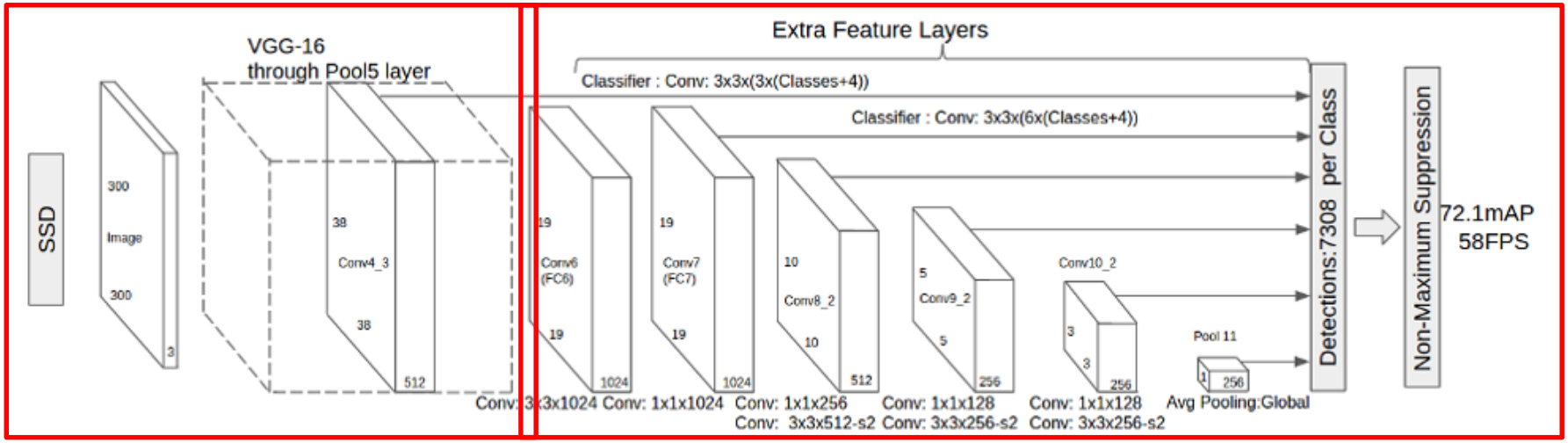
참고: 'Semantic Image Segmentation with deep convolutional nets fully connected CRFs'

(2) Extra Feature Layer

① Multi-scale feature maps for detection

② Convolutional predictors for detection

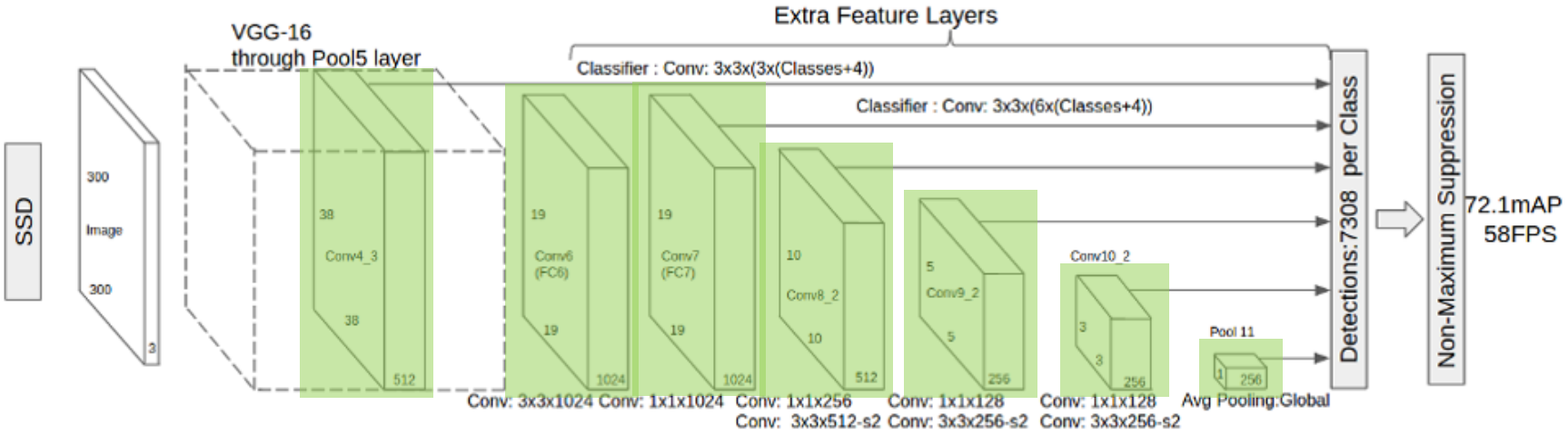
③ Default boxes and aspect ratios



2.1 Model - Extra Feature Layer

① Multi-scale feature maps for detection

- Feature Layer size가 점점 줄어들어서 multiple scales 에서 detection 가능
- 각 feature map에 적용된 classifier는 다른 convolutional model 사용
- (↔ Overfeat, YOLO: single scale feature map에서 수행)



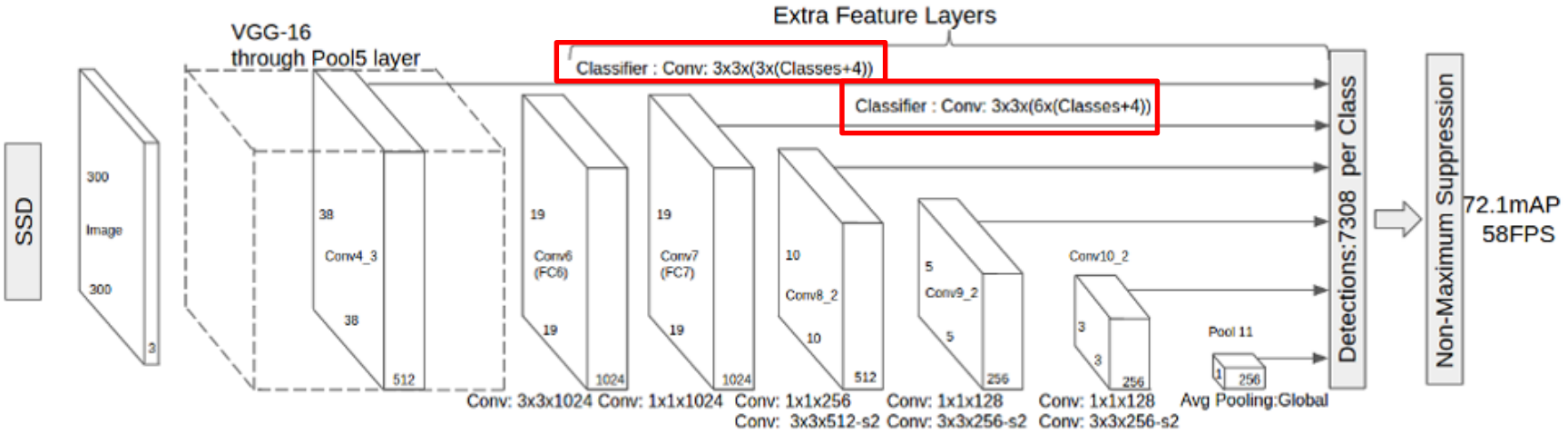
2.1 Model - Extra Feature Layer

② Convolutional predictors for detection

- *m x n x p* 크기의 feature map에 *3 x 3 x p* 크기의 작은 filter 사용

  → 하나의 ①score(for category) ②shape offset relative to default box 좌표

- (↔ YOLO : convolutional filtering 단계 대신, fully connected layer 있음)



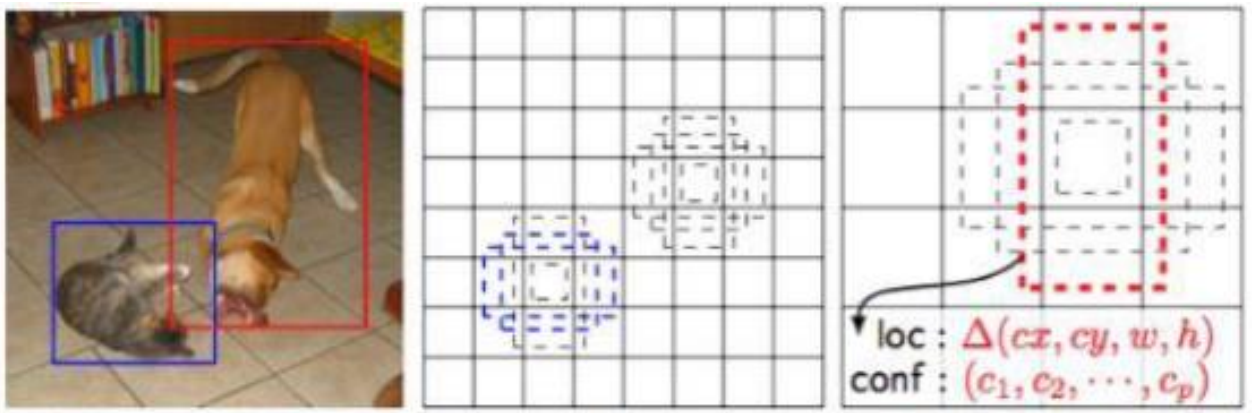
2.1 Model - Extra Feature Layer

③ Default boxes and aspect ratios

- GT box(빨/파) 정보→ default box offset 값 저장
- (Faster R-CNN의 anchor box와 비슷한 개념)
- Default box 계산 *(e.g. K=4)*

e.g. feature map 크기: *m x n* / class # : *c* / 하나의 위치에서 default box # : *k*

→ **output # : *(c+4)*k*m*n***



(a) Image with GT boxes (b) 8 × 8 feature map (c) 4 × 4 feature map

2.2 Training

- Region Proposal 방법과 SSD 방법 Key difference
  : Ground Truth가 여러 feature map output에 적용됨

- **Training Methods**
  ① Matching Strategy
  ② Training objective
  ③ Choosing scales and aspect ratios for default boxes
  ④ Hard negative mining
  ⑤ Data augmentation

2.2 Training

① Matching Strategy

- Ground Truth에 맞는 default box 미리 결정
  → Jaccard overlap(IoU) > 0.5 (Multi box 논문 방법과 동일)
  → 0.5 이상: 여러 scale에서 여러 매칭 허용
- (↔ Multibox 에서는 Ground Truth와 일치하는 하나의 default box만 찾음)

② Training objective

Class score: Softmax **Localization loss:** Soft L1 loss

- (Faster R-CNN 과 유사)

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (1)$$

where N is the number of matched default boxes, and the localization loss is the Smooth L1 loss [6] between the predicted box (l) and the ground truth box (g) parameters. Similar to Faster R-CNN [2], we regress to offsets for the center of the bounding box and for its width and height. Our confidence loss is the softmax loss over multiple classes confidences (c) and the weight term α is set to 1 by cross validation.

2.2 Training

③ Choosing scales and aspect ratios for default boxes

- Single Network에서 Scale 문제 해결
- (↔ Overfeat, SPPnet: Pyramid 구조를 이용하여 scale 문제 해결)

e.g. feature map # : m / Smax = 0.95 / Smin = 0.2

→ **단순 비율 증가**

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m - 1}(k - 1), \quad k \in [1, m] \tag{2}$$

We impose different aspect ratios for the default boxes, and denote them as $a_r \in \{1, 2, 3, \frac{1}{2}, \frac{1}{3}\}$. We can compute the width ($w_k^a = s_k\sqrt{a_r}$) and height ($h_k^a = s_k/\sqrt{a_r}$) for each default box. For the aspect ratio of 1, we also add a default box whose scale is $s'_k = \sqrt{s_k s_{k+1}}$, resulting in 6 default boxes per feature map location. We set the center of each default box to $(\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|})$, where $|f_k|$ is the size of the k-th square feature map, $i, j \in [0, |f_k|)$, and we truncate the coordinates of the default boxes such that they are always within [0, 1]. In practice, one can also design a distribution of default boxes to best fit a specific dataset.

2.2 Training

④ Hard negative mining

- 위의 Matching 을 거치면 대부분의 데이터는 negative sample 임.
- Default box에 대하여 confidence 높은 순으로 정렬 후,
  negative : positive = 3:1 로 정렬

⑤ Data augmentation

① 원본 이미지 그대로 사용

② 최소 jaccard 비율이 0.1, 0.3, 0.5, 0.7, 0.9인 patch를 random 선택

③ Random sampling

- Sample patch 크기는 원본 크기의 [0.1 1] 비율로 결정.
- Aspect ratio는 ½ 와 2 사이에서 결정
- 이렇게 추출된 patch는 fixed size 300x300 또는 500x500으로 변환되고 0.5 확률로 horizontal flip 함.

Experimental Results

- Data augmentation is crucial
- More feature map is better
- More default box shapes is better
- Atrous is better and faster

(; VGG16 사용. pool5 그대로 두고 fc6, fc7에서 subsampling 안하고 conv5_3을 prediction map에 추가
→ 0.7% 성능 떨어지고 50% 느려짐)

Experimental Results

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	**74.8**	80.4	70.4
Faster [2]	73.2	76.5	79.0	70.9	**65.5**	**52.1**	83.1	84.7	86.4	52.0	**81.9**	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
SSD300	72.1	75.2	**79.8**	70.5	62.5	41.3	81.1	80.8	86.4	51.5	74.3	**72.3**	83.5	84.6	80.6	74.5	46.0	71.4	73.8	83.0	69.1
SSD500	**75.1**	**79.8**	79.5	**74.5**	63.4	51.9	**84.9**	**85.6**	**87.2**	**56.6**	80.1	70.0	**85.4**	**84.9**	**80.9**	**78.2**	**49.0**	**78.4**	72.4	**84.6**	**75.5**

Table 1: PASCAL VOC2007 `test` **detection results.** Both Fast and Faster R-CNN use input images whose minimum dimension is 600. The two SSD models have exactly the same settings except that they have different input sizes (300×300 vs. 500×500). It is obvious that larger input size leads to better results.

Method	*mAP*	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	**89.3**	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
Faster [2]	70.4	**84.9**	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	**77.1**	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
YOLO [5]	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
SSD300	70.3	84.2	76.3	69.6	53.2	40.8	78.5	73.6	88.0	50.5	73.5	**61.7**	85.8	80.6	81.2	77.5	44.3	73.2	**66.7**	81.1	65.8
SSD500	**73.1**	**84.9**	**82.6**	**74.4**	**55.8**	**50.0**	**80.3**	**78.9**	88.8	**53.7**	76.8	59.4	**87.6**	**83.7**	**82.6**	**81.4**	**47.2**	**75.5**	65.6	**84.3**	**68.1**

Table 3: PASCAL VOC2012 `test` **detection results.** Fast and Faster R-CNN use images with minimum dimension 600, while the image size for YOLO is 448×448.

Method	data	Average Precision		
		0.5	0.75	0.5:0.95
Fast R-CNN [6]	train	35.9	-	19.7
Faster R-CNN [2]	train	42.1	-	21.5
Faster R-CNN [2]	trainval	42.7	-	21.9
ION [21]	train	42.0	23.0	23.0
SSD300	trainval35k	38.0	20.5	20.8
SSD500	trainval35k	43.7	24.7	24.4

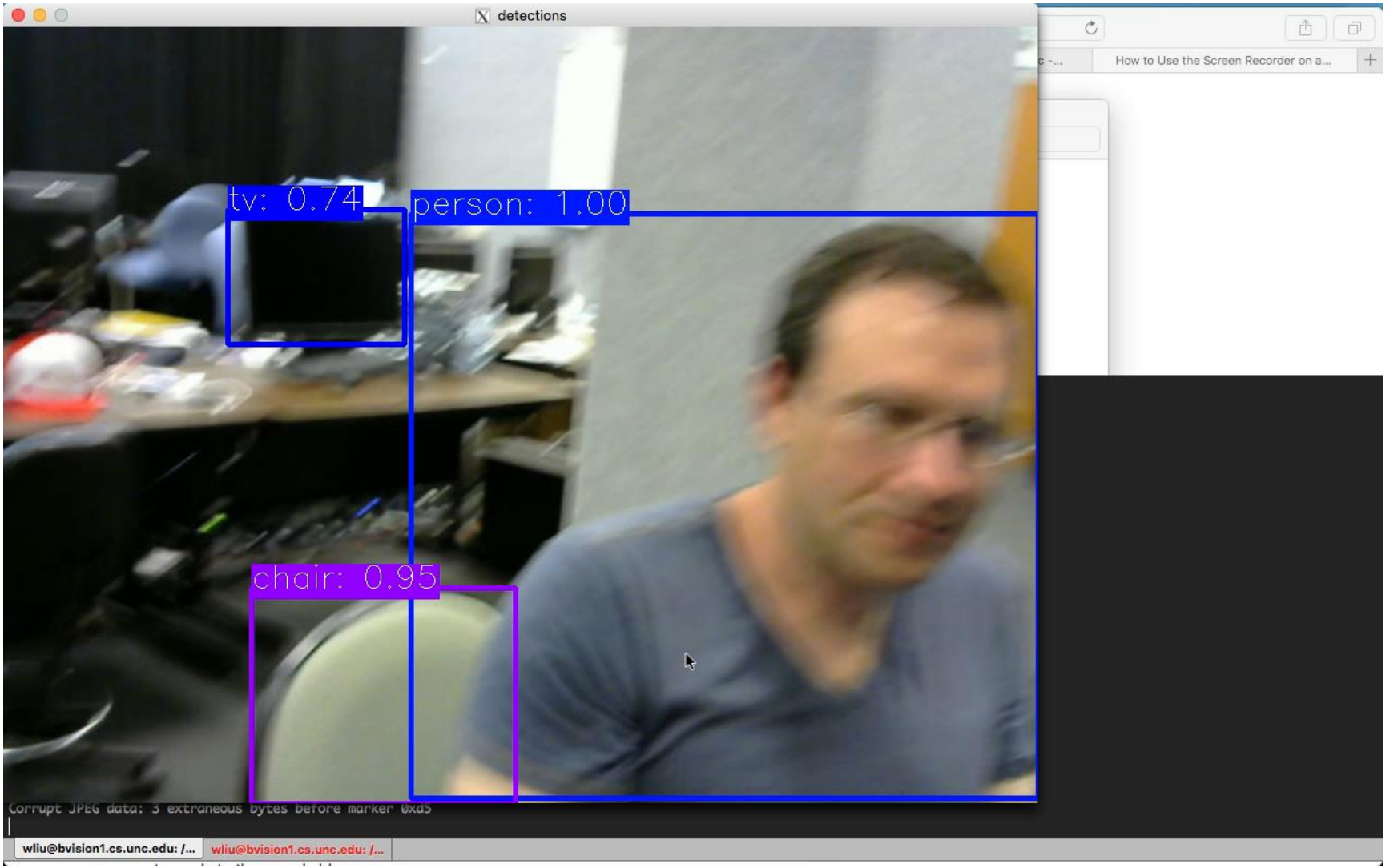
Table 4: MS COCO `test-dev2015` **detection results.**

Method	*mAP*	**FPS**	**# Boxes**
Faster R-CNN [2](VGG16)	73.2	7	300
Faster R-CNN [2](ZF)	62.1	17	300
YOLO [5]	63.4	45	98
Fast YOLO [5]	52.7	155	98
SSD300	72.1	58	7308
SSD500	**75.1**	23	20097

Table 5: **Results on Pascal VOC2007** `test`**.** SSD300 is the only real-time detection method that can achieve above 70% mAP. By using a larger input image, SSD500 outperforms all methods on accuracy while maintaining a close to real-time speed. The speed of SSD models is measured with batch size of 8.

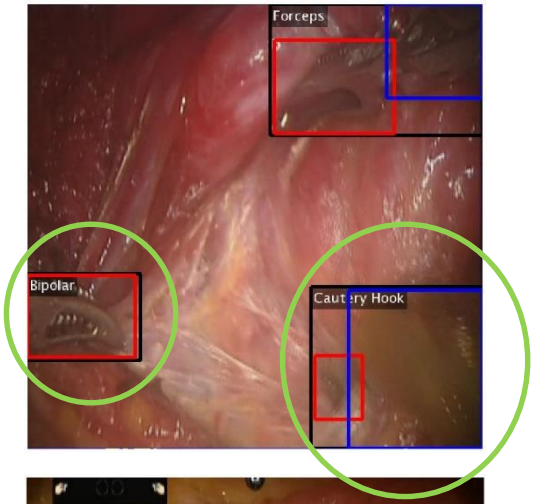
Fig. 5: Detection examples on MS COCO `test-dev` with SSD500 model. We show detections with scores higher than 0.6. Each color corresponds to an object category.

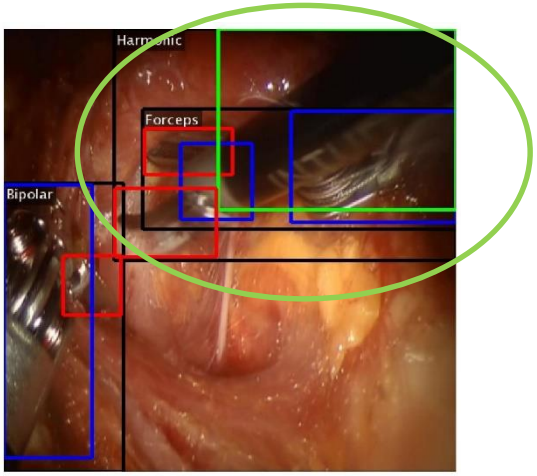
Experimental Results (Demo)

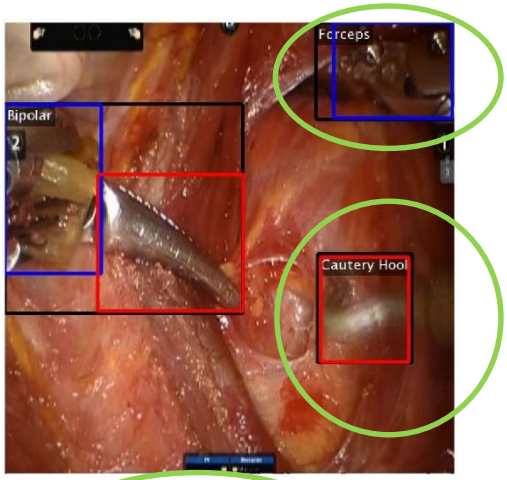


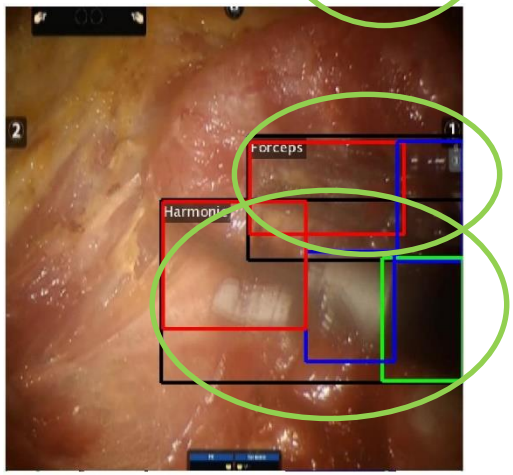
Issues

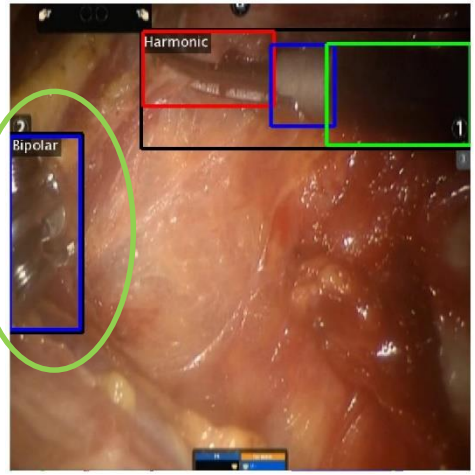
※ Part of View, Blurry, Occlusion

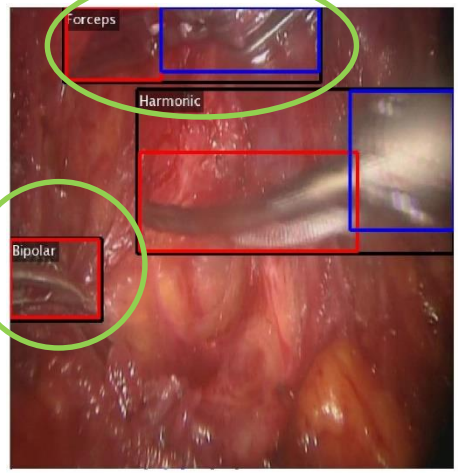












감사합니다

Appendix

Faster R-CNN: Region Proposal Network

Use **N anchor boxes** at each location

Anchors are **translation invariant**: use the same ones at every location

Regression gives offsets from anchor boxes

Classification gives the probability that each (regressed) anchor shows an object