Ice Hockey Video: Improving Camera Movements

Jason Durr
Justin Johnson
Chang Ju Kim
Adam Wipprecht

Current State of Research

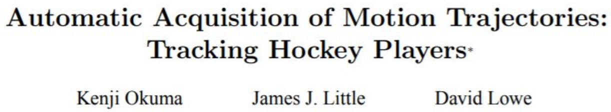
- Many previous similar studies, with slight variations

Fast and Reliable Detection of Hockey Players ☆

Iveta Mrazova, Matej Hrincar

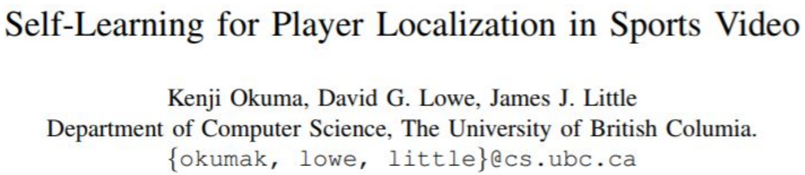
Learning to Track and Identify Players from Broadcast Sports Videos

Wei-Lwun Lu, Jo-Anne Ting, James J. Little, Kevin P. Murphy



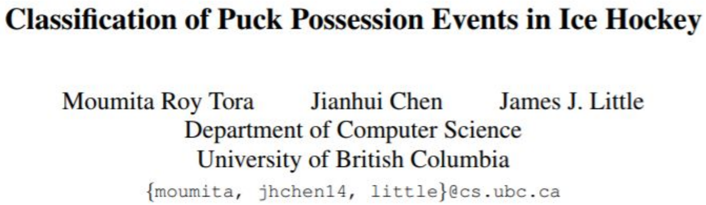
Automatic Acquisition of Motion Trajectories: Tracking Hockey Players*

Kenji Okuma James J. Little David Lowe



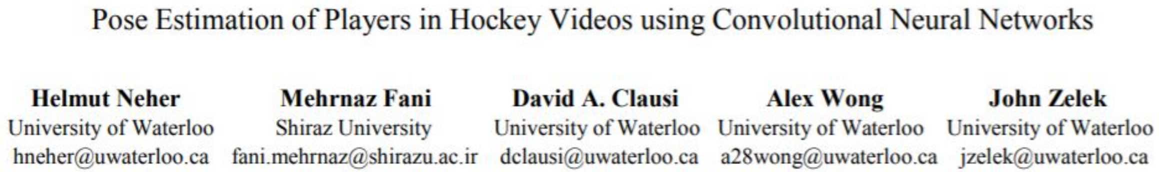
Self-Learning for Player Localization in Sports Video

Kenji Okuma, David G. Lowe, James J. Little
Department of Computer Science, The University of British Columia.
{okumak, lowe, little}@cs.ubc.ca



Classification of Puck Possession Events in Ice Hockey

Moumita Roy Tora Jianhui Chen James J. Little
Department of Computer Science
University of British Columbia
{moumita, jhchen14, little}@cs.ubc.ca



Pose Estimation of Players in Hockey Videos using Convolutional Neural Networks

Helmut Neher	Mehrnaz Fani	David A. Clausi	Alex Wong	John Zelek
University of Waterloo	Shiraz University	University of Waterloo	University of Waterloo	University of Waterloo
hneher@uwaterloo.ca	fani.mehrnaz@shirazu.ac.ir	dclausi@uwaterloo.ca	a28wong@uwaterloo.ca	jzelek@uwaterloo.ca

Tools Utilized

- Darknet / YOLOv3
- OpenCV
- Hardware:
    - Intel I9-9900X
    - RTX 2070

What We Have Done - Data Collection

- To train a custom model you need a large data set ... Or so we thought
  - Accuracy is prefered to size
  - First tried 40060 single player images
  - Next tried to use 7000 semi-accurate samples
  - Final model uses 317 hand annotated image

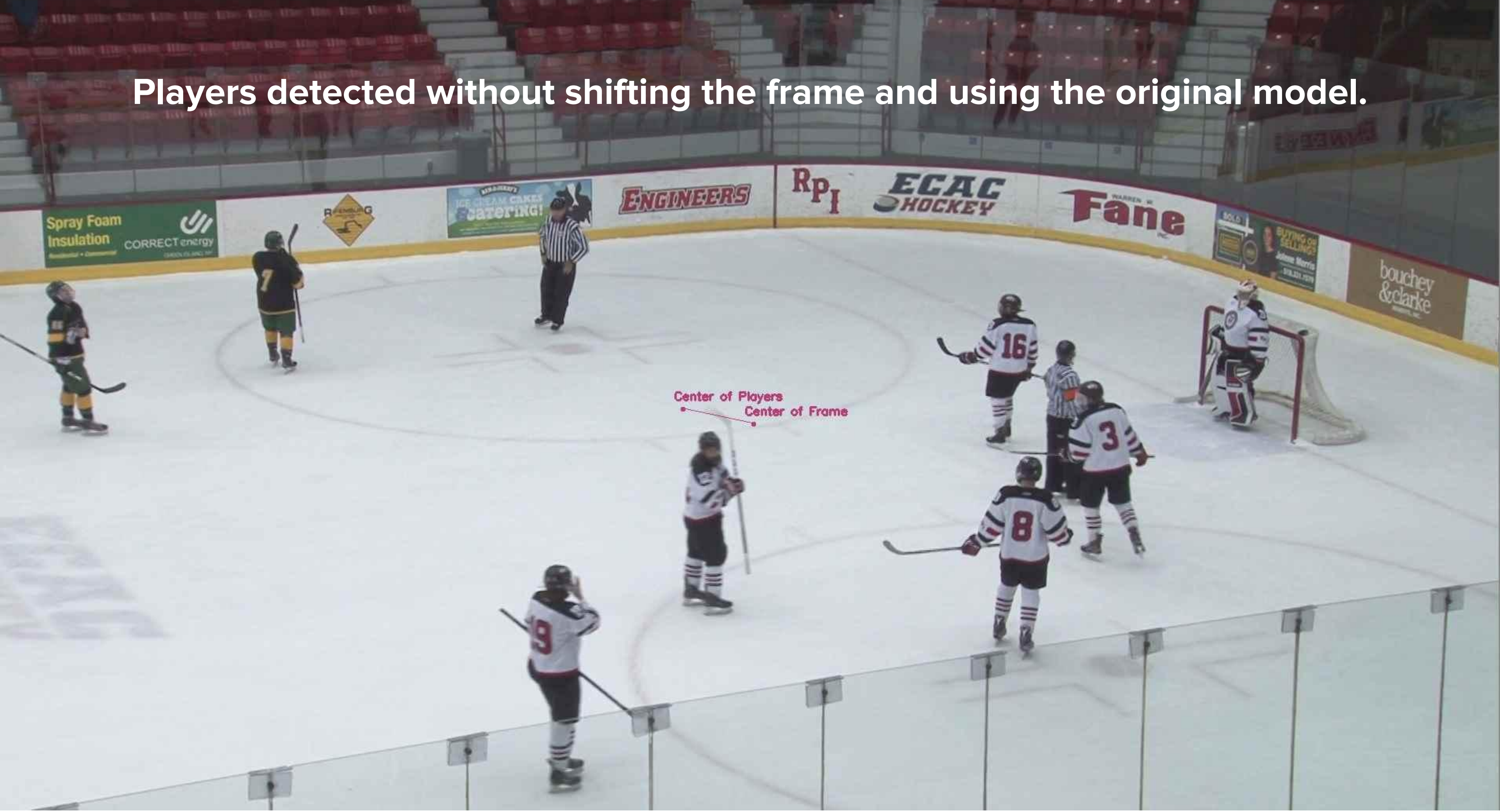
What We Have Done - Model Training

- Training the model using Darknet
  - Open source NN framework written in C and CUDA
  - GPU acceleration is necessary for model training
  - Trained several models on various datasets
  - Based on the You Only Look Once, YOLO, object detection algorithm
  - Best results between 20,000-30,000 iterations

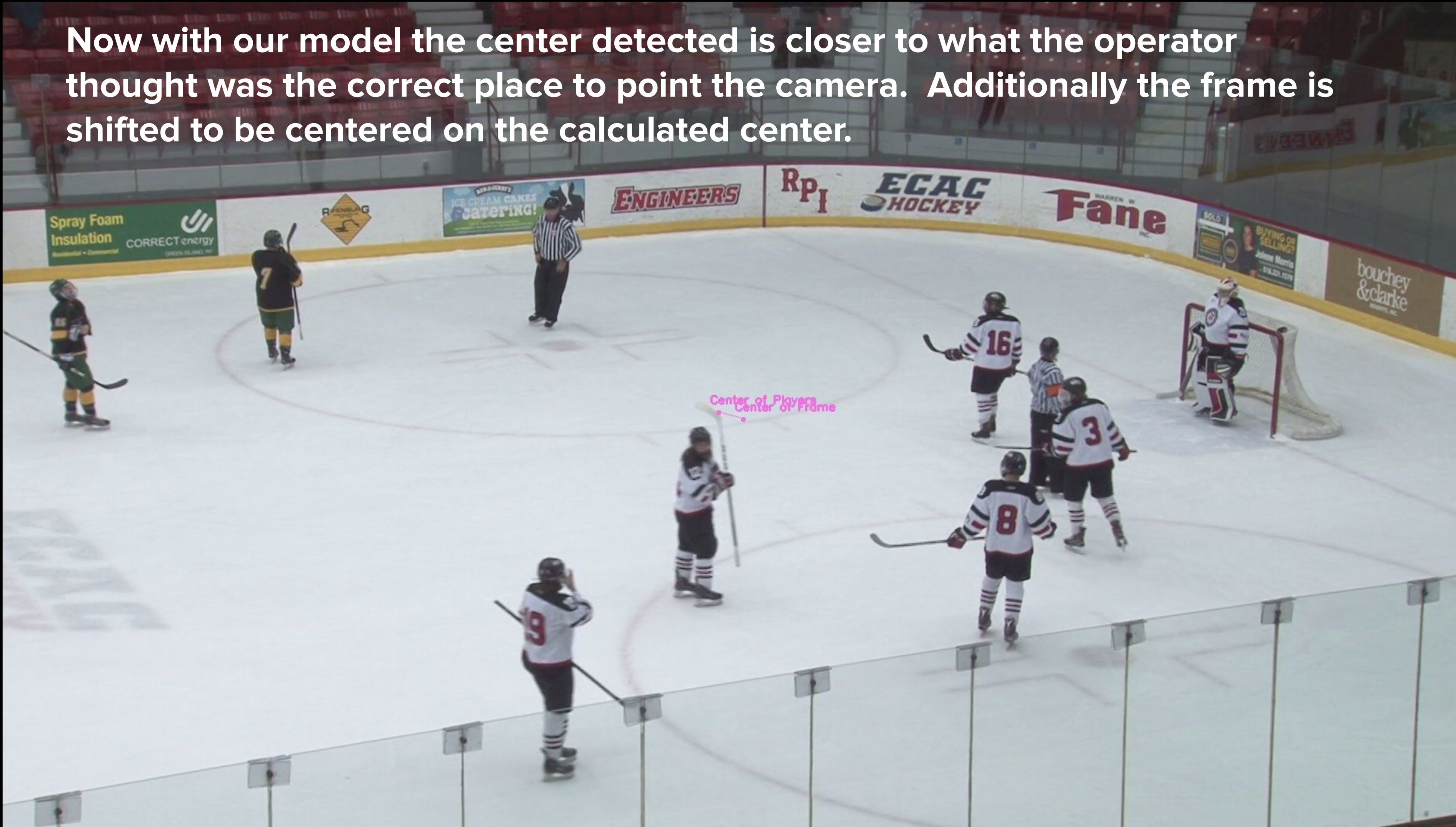
What We Have Done - Algorithms

- To determine where to point the camera we chose to point it towards the average location in the frame of where the players are in the frame
  - The idea is the camera should always be pointed towards center of where the majority of the players are and by centering on the average there should be the most possible players in the frame
  - Obviously this has some drawbacks as there are a lot of players at center ice when they are doing shift changes as well as when there is a fast break the camera might not follow in the right direction

Players detected without shifting the frame and using the original model.



Now with our model the center detected is closer to what the operator thought was the correct place to point the camera. Additionally the frame is shifted to be centered on the calculated center.



Extensions and Future Goals

- Try out other algorithms for choosing where to point on the Ice
  - Currently the camera just tries to point at the average location of all the players on the ice but other algorithms might be better
  - One algorithm to try would be looking at two frames to guess the direction the players are moving and pointing to where they will be

Extensions and Future Goals

- Speed up the recognition algorithm
  - The current method to track takes about 1 second to run on a laptop and half a second to run on a 10 core 20 thread computer. In order to be a practical solution it would need to only take a few frames to find the people otherwise there would be a significant lag in filming
  - In order to achieve this we could try to decrease the size of the neural net to increase speed until the gain in speed is no longer worth the decrease in accuracy

Extensions and Future Goals

- Implement hardware that is able to move the camera
  - Currently the software is only able to say where the ideal center of frame would be and a hardware solution that would move the camera to point at that location would need to be implemented

More Progress

- Currently also runs on GPU
  - Bypassed CPU bottleneck which made it run faster
- Tried multithreading
  - Program ran faster but did not scale as well as we hoped