# Video Annotation and Tracking with Active Learning

Carl Vondrick Deva Ramanan Department of Computer Science University of California, Irvine

"Maybe it's more bizarre that I keep doing these hits for a penny. I must not be the only one who finds them oddly compelling–more and more boxes show up on each hit." — Anonymous subject

**Unsolved Problem 1**: No fully automatic solution today is capable of detecting and tracking both the people and the basketball.

**Unsolved Problem 2**: Building large video data sets is inefficient because frame-by-frame hand labeling is slow, costly, and tedious.

## Motivations

1. What is the best division of labor for crowdsource video labeling?
2. What are the tradeoffs between automation and manual labeling?
3. Given a fixed budget, what is the best accuracy we can achieve?

## Contributions

- A set of "best-practices" for crowdsourced video annotation.
- In contrast to [3], can interpolate nonlinear paths w/o much effort.
- Expanding [2] to analyze tradeoffs between human and CPU cost.
- Ability to build *massive* video data sets under a budget.
- A reusable, open source video annotation platform for affordable, research video labeling.

## Mechanical Turk

- Mechanical Turk: online, monetized, crowdsourced marketplace.
- Ideal for tasks that are hard for computers, but trivial for humans.
- Workers complete *Human Intelligence Tasks* and we get results.

## The "Turk Philosophy"

- Suggests completely replacing automation with human effort.
- For *Images*: annotate every object (highly successful). [1]
- For *Video*: hand label every frame (highly inefficient).
- Given the redundant yet dynamic nature of video, we need an approach that combines the computational power of the CPU with the superior vision capability of humans.

## References


[1] A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. *Urbana*, 51:61820, 2008.

[2] S. Vijayanarasimhan and K. Grauman. Whats It Going to Cost You?: Predicting Effort vs. Informativeness for Multi-Label Image Annotations. CVPR, 2009.

[3] J. Yuen, B. Russell, C. Liu, and A. Torralba. LabelMe video: Building a Video Database with Human Annotations. 2009.


## Interactive Video Player

- Browser video player that guides a worker to label an entity.
- First, instructs user to draw a box around an item of interest.
- User then adjusts the box when video pauses on the next key frame.

- Extracted into frames: removed artifacts from Flash video codec.
- Carefully manage frame caching to reduce bandwidth.
- Wider participation across platforms without Flash support.

## Quality Assurance

- No quality guarantee. Workers motivated to finish quickly.
- Experiments indicate **35%** of labels were poor (see below).
- Identify degenerates through hand validation, statistical overlap, heuristic technique, or user agent identification string.

## Dense Labeling Protocol

1. Worker instructed to annotate an unlabeled entity.
2. If initial frame fully labeled, advance to next key frame.
3. Instructs worker to label again — repeat (2) if still none.
4. If worker can track and work is not degenerate, add to video.
5. Else, if no new objects are discovered, vote to finish.
6. After enough votes, server stops spawning HITs for the video.

## Video Server Cloud

- Server written in Python 2.6 and Cython. Client in JavaScript.
- Entirely open source. Can deploy to clouds without license fees.

A year's worth of experiments. Workers produced quality annotations throughout the 210,000 frame basketball game.

## Linear Interpolation

- The simplest tracking approach is linear interpolation:

$$b_t^{lin} = \left(\frac{t}{T}\right) b_0 + \left(\frac{T-t}{T}\right) b_T \quad \text{for} \quad 0 \leq t \leq T$$

- But, objects do not necessarily move linearly and can be chaotic.

## Discriminative Object Templates

- Extract both HOG and RGB histogram from foregrounds and backgrounds from the annotated frames:

$$\phi_n(b_n) = \begin{bmatrix} HOG \\ RGB \end{bmatrix} \qquad y_n \in \{-1, 1\}$$

- Learn a SVM weight vector, $w$, that minimizes a linear loss:

$$w^* = \operatorname{argmin} \frac{1}{2} w \cdot w + C \sum_n^N \max(0, 1 - y_n w \cdot \phi_n(b_n))$$

- Data is very complex. Simpler templates perform poorly.

Can you spot all the difficult objects?

## Constrained Tracking

- Calculate a least cost path between constrained endpoints:

$$\underset{b_{1:T}}{\operatorname{argmin}} \sum_{t=1}^{T} U_t(b_t) + P(b_t, b_{t-1})$$

- Local cost is SVM score plus linear deviation, but truncated:

$$U_t(b_t) = \min\left(-w \cdot \phi_t(b_t) + \alpha_1 ||b_t - b_t^{lin}||^2, \alpha_2\right)$$

- Pairwise cost ensures path is smooth and does not teleport:

$$P(b_t, b_{t-1}) = \alpha_3 ||b_t - b_{t-1}||^2$$

- Dynamic programming efficiently solves the recursion:

$$cost_0(b_0) = U_0(b_0)$$
$$cost_t(b_t) = U_t(b_t) + \min_{b_{t-1}} cost_{t-1}(b_{t-1}) + P(b_t, b_{t-1})$$

## Experiments & Results

- Diminishing returns: each click has less impact than previous.
- The "Turk philosophy" is *not* efficient for video.
- Trade-off between CPU cost vs. human cost for max. accuracy.
- A visual based tracker *can* benefit video annotation.
- Interactive vision: with modest human effort, we *can* deploy algorithms that quantify progress in difficult scenarios.