Ball Tracking, Court Calibration and Point Segmentation for Amateur Tennis Video on a Phone

Jacob Beal

Technical report · November 2026 · Independent R&D for MyBreakPoint, not peer reviewed
© 2026 Break Point Technologies, LLC. All rights reserved.

Abstract. Amateur tennis recordings combine off-center cameras, small motion-blurred targets, variable lighting, and long intervals without play. This report studies an on-device system integrating a five-frame heatmap ball detector with candidate-level tracking, a court-keypoint network with robust homography estimation, and a temporal segmenter using ball, motion, and audio features. The detector and tracker localize 218 of 248 visible-ball frames within 8 pixels at 512×288 resolution (87.9%), with 20 false alarms on 88 no-ball frames across seven videos; this evaluation excludes court gating. On 359 amateur frames from ten videos, domain adaptation and temporal aggregation increase the fraction within 15 cm court-plane error from 25.1% to 61.9%, measured on a checkpoint-selection validation set. Across 42 videos and 1,351 labeled serve segments, held-fold decode re-selection retains 91.9% of segments while removing 52.4% of dead time on average; retention ranges from 91.9% to 92.4% over three training seeds. This is a sensitivity analysis on existing predictions, not fully nested model cross-validation. The study documents domain shift, compression sensitivity, overlapping-window leakage, and timestamp-alignment errors. Small correlated samples, single-reviewer labels, residual selection effects, and withheld experimental assets limit generalization and independent reproduction.

---

1 Introduction

MyBreakPoint is a tennis app for teams and players. I am its only developer. Two of its features depend on computer vision:

- **Live ball tracking** overlays the ball and court while a match is streamed.
- **Points only** cuts a recorded match down to the points, removing walking, ball collection and changeovers.

Both run on the phone, offline, which rules out large models and server inference.

The input is amateur footage: a phone on a fence, bench or tripod behind or beside the court, at 360p to 1080p, in daylight, dusk, indoor halls or under floodlights. Broadcast datasets, which most published tennis models are trained on, look quite different: elevated centered cameras, consistent lighting and sharp video. One result below makes the difference concrete. A court model with 97.5% accuracy on broadcast frames scored 25.1% on phone footage.

This report covers the three models in Figure 1 and how they were evaluated:

1. **Ball tracking** (Section 3): a 5-frame heatmap network and a tracker that chooses among local maxima instead of trusting each frame's argmax.
2. **Court calibration** (Section 4): a 14-keypoint network, a robust fit to the court model in meters, and a gate that withholds ball positions when the court is not reliably in view.

3. **Point segmentation** (Section 5): a temporal convolutional network over ball, motion and audio features, decoded into playable segments.
4. **Evaluation methodology** (Section 6): group-aware splits, cross-validation, and an account of what made earlier numbers unreliable.

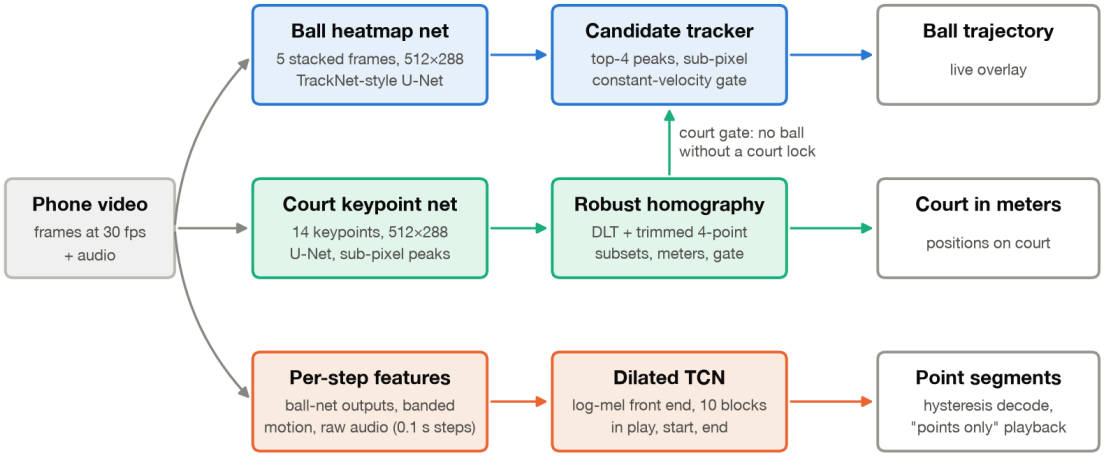


Figure 1: The three models and how they connect. Everything runs on the phone. Ball positions are only reported while the court gate holds a court lock.

The code is published for reading. The footage, labels, weights and tuned operating values are withheld (Section 10). Quantitative results trace to retained evaluation artifacts; historical observations without recoverable logs are labeled separately; the repository's `docs/claims.md` lists each one with its source.

2 Contributions and related work

The contribution is an integrated system and an empirical study of amateur tennis video: candidate-level temporal tracking, court-plane validation, and multimodal point segmentation under mobile deployment constraints. The evaluation identifies domain shift, compression sensitivity, timestamp-alignment errors, and model-selection bias. It does not introduce a new general-purpose network family or establish state-of-the-art performance.

TrackNet and TrackNetV2 motivate multi-frame heatmap tracking [1], [2]. WASB-SBDT provides the pretrained teacher and local comparison [3]. More recent motion-attention work, TrackNetV4 [4], is relevant but was not evaluated here; no ranking against it is claimed. TennisCourtDetector supplies the public court-keypoint layout and data [5], while temporal convolutional networks provide the segmentation model family [6]. Published benchmark scores are not directly comparable to this report's labels, metrics, and camera distribution.

3 Ball tracking

3.1 Model

The detector follows the TrackNet idea [1], [2] in an independent implementation. Five consecutive RGB frames are resized to 512×288 with area interpolation, stacked to 15 channels, and passed through a VGG-style encoder. The encoder has four stages of 2, 2, 3 and 3 convolutions with base width 16. A decoder with bilinear upsampling and skip connections produces one heatmap for the last frame.

Training uses a Gaussian-target focal loss. The output bias is initialized to a 1% prior, because with a zero bias the summed focal loss starts with very large gradients.

3.2 From heatmaps to a trajectory

Accepting each frame's argmax above a fixed threshold throws away two kinds of information:

- When the argmax lands on a shoe or a line marking, the ball is often still a secondary peak.
- The ball moves smoothly, so its recent path predicts where it is now.

The tracker therefore works like this:

1. Take the top 4 local maxima per frame and refine each to sub-pixel by a local centroid.
2. A constant-velocity track starts only after consecutive confident detections.
3. Once confirmed, the track accepts a weaker peak inside a gate around its prediction. The gate widens for each frame the track coasts without a detection, and the track ends after a few empty frames.

Distances are fractions of the input width, so one configuration works at any model resolution. The tuned confidence levels and gate sizes are withheld.

3.3 Evaluation

Ground truth comes from 350 frames, 50 from each of 7 test videos covering day, dusk and night play. Frames were sampled uniformly over in-play time with a fixed seed, independent of any model. Each frame is labeled `visible` (with a position), `no ball`, `not in play` or `uncertain`. Candidate positions were proposed by the evaluated model and by a teacher model (WASB-SBDT [3]), then reviewed. 125 of the 248 visible positions were placed without a matching candidate.

The score measures the **detector and tracker without the court gate**. It is not a whole-app accuracy measurement. For each labeled frame, the model and tracker run causally over the preceding 1.5 s at 30 fps, and the tracker's output at that frame is compared with the label:

- On a visible frame, a hit is an output within 8 px (and 3 px) of the label, in 512×288 model pixels, about 10 to 13 px in the 360p/480p source. No output counts as a miss.
- Any output on a no-ball or not-in-play frame is a false alarm. Uncertain frames are excluded.

Evaluation files were acquired after training, but this does not prove source-video disjointness. The full training manifest for these weights was not retained, so overlap through upstream datasets cannot be ruled out.

3.4 Results

System	Within 8 px	Within 3 px	False alarms
Shipped model + tracker	**218 / 248 (87.9%)**	204 (82.3%)	20 / 88
WASB-SBDT teacher, same tracker (cached candidates)	210 (84.7%)	202 (81.5%)	28 / 88
WASB-SBDT teacher, end-to-end pipeline	205 (82.7%)	193 (77.8%)	33 / 88

Table 1: Ball tracking system score on 248 visible-ball frames and 88 no-ball frames from 7 test videos. The 95% Wilson interval for the shipped model's 8 px rate is 83.3% to 91.4%.

The shipped model reaches 87.9% within 8 px (Table 1):

- **By lighting:** 93.8% on the three day videos, 77.1% on the dusk video and 85.1% on the three night videos. The lighting tags are rough; one night-tagged video looks like daylight.
- **By video:** from 77.1% to 97.6%.

- **Against the teacher:** the shipped model is ahead on 21 frames and behind on 13. The difference is not significant (exact McNemar test, $p = 0.23$). This compares the cached teacher row, not the separately decoded teacher. Observed localization and false-alarm counts favor the shipped model on this sample; neither equivalence nor superiority is established. The frame-level test and Wilson interval ignore within-video correlation.

Pooling both models' candidates and smoothing them with a second-order Viterbi pass raised the 8 px rate to 89.9% to 91.5%, but with 33 to 58 false alarms instead of 20. These settings were tuned on the same 350 frames, so even that gain is optimistic. The app does not use pooling.

A bug found while preparing this report. Label times are stored rounded to the millisecond, and the original full-pipeline scoring script matched labels to frames with a strict 0.1 ms comparison. That scored 67 of the 248 visible frames one frame late. Its only saved log reported 82.7% overall, and the figure in the project notes ("88%") was the day-only subset of that log. Matching each label to the nearest frame gives 218/248 (87.9%) through the full pipeline, exactly the figure obtained independently from cached candidates. A second script, used for the format experiments, had the mirror-image bug (59 frames scored one frame early). Those experiments were rerun.

3.5 Format and quality

A separate on-device ONNX artifact, with a different SHA-256 from the headline ball artifact, was run without court gating on six test videos re-encoded from a higher-resolution source (217 visible frames, one run each, fixed frame matching):

Variant	Within 8 px, bilinear	Within 8 px, area
Original 360p/480p	192 (88.5%)	192 (88.5%)
High-res source (720p/1080p)	192 (88.5%)	193 (88.9%)
720p re-encode	196 (90.3%)	193 (88.9%)
Heavy compression (CRF 36)	**147 (67.7%)**	**138 (63.6%)**
25 fps	185 (85.3%)	183 (84.3%)
Synthetic darkening	176 (81.1%)	181 (83.4%)

Table 2: Ball tracking under format changes, six videos, 217 visible frames.

Heavy compression is the only large and consistent effect (Table 2): 45 fewer frames within 8 px, worse on all six videos, mostly from missed detections.

- **Resolution:** no detectable effect.
- **Frame rate:** 25 fps is within noise at 8 px.
- **Darkening:** costs about 16 frames, unevenly across videos.
- **Resize method:** area versus bilinear shows no consistent difference.

Two earlier conclusions, that high-res footage tracked worse and that 25 fps beat high-res, were artifacts of the timing bug.

4 Court calibration

4.1 Model and geometry

A U-Net style network (base width 32) predicts heatmaps for 14 court keypoints at 512×288: the baseline corners, the singles sideline ends, the service line ends and the center service line ends. The layout follows the TennisCourtDetector dataset [5]. Peaks are refined to sub-pixel.

The court model uses ITF dimensions in meters, with the origin at the center of the net (Figure 2). A normalized direct linear transform estimates the image-to-court homography. A robust variant repeatedly drops the worst point while any court-plane error exceeds a threshold. It then tries every 4-point subset, keeps the largest consensus set within 0.30 m and refits. At least 6 inliers are required. Errors are measured on the court plane in meters, so a far-court keypoint counts by its real distance, not its pixel distance.

A court gate decides whether any ball position is trusted. It requires enough keypoints, low reprojection error, a convex court quadrilateral of plausible size and consecutive detections, and a lock expires after a short time.



Figure 2: Court calibration on a frame from the author's match. Left: aggregated keypoint detections (orange: inliers; red: detected but rejected by the robust fit; two corners are out of frame) and the court model projected through the fitted homography. Right: the court coordinate system in meters, origin at the net center, x across the court and y toward the far baseline.

4.2 Evaluation and results

For each frame, a 12×24 grid of court points is projected through the labeled homography and mapped back through the predicted one. The frame's error is the median distance on the court plane, and a frame passes if it is within 15 cm. Frames with no homography count as failures.

Model	Evaluation set	Within 15 cm	Median error
Broadcast-trained	200 broadcast frames	97.5%	4.2 cm
Broadcast-trained	359 amateur frames, 10 videos	25.1%	604 cm
Amateur-trained, single frame	same 359 frames	53.5%	
Amateur-trained, 12-frame median	same 359 frames	**61.9%**	12.2 cm

Table 3: Court calibration accuracy. The amateur set was also used to select checkpoints, so the amateur-trained rows are validation results. The broadcast set shares 434 of its 442 source videos with its dataset's training split.

The broadcast-trained model looked excellent on broadcast frames and failed on phone footage (Table 3). Because ball tracking waits for a court lock, this explained most of the poor live tracking seen in the app. Training on amateur frames more than doubled the amateur score, and the app's

running median over 12 detections adds about 8 points more. Four of the ten amateur videos still fail completely.

On sharper 1080p footage of one view, seen through the net mesh, accuracy fell from 88% to 32%. That model had only seen 480p training frames.

5 Point segmentation

5.1 Features and model

The video is cut into steps of 0.1 s. Each step has 8 visual features:

- five from the ball model run exactly as on the phone: peak confidence, the share of frames with a confident ball, ball speed, and mean position;
- three motion scores: brightness-normalized frame differences in three horizontal bands of a 64×36 thumbnail, roughly the far court, the net area and the near court.

The motion bands let a far-side serve register even when the ball is too small to track, and normalizing by brightness keeps night and day comparable.

Audio enters the network as raw 16 kHz samples. The log-mel front end (64 bands, 25 ms window, 10 ms hop) is a fixed convolution inside the model, so training and the phone compute the same spectrogram.

The network:

- an audio encoder (48 dimensions per step) and a feature MLP (32);
- a local convolution to 64 channels;
- ten residual dilated temporal blocks with dilations 1, 2, 4, 8, 16, 32, 64, 1, 4 and 16, in the style of temporal convolutional networks for action segmentation [6], seeing about 15 s on each side of a step.

Three output channels predict in play, point start and point end.

Training uses per-step binary cross-entropy on in-play plus a boundary term (weight 0.5), 60 s random windows, 1,500 steps of batch 16 and learning rate 2×10^{-3}. Reviewed segments of 4 s or less (serve faults, unreturned serves) get 3× weight, because per-step losses otherwise barely register them. Videos with only automatic labels get weight 0.15.

Decoding applies hysteresis thresholds, a minimum segment length and merging of short gaps. The thresholds are rescaled per video by that video's own probability distribution, which helps on videos where the model is under-confident. Playback pads each segment by 1.0 s before and 2.0 s after and merges gaps under 2.5 s.

5.2 Metrics

- **Points kept whole:** a labeled serve segment (point, fault or let) counts if at least 95% of it, on a 0.05 s grid, is inside the padded playback. Pass or fail, pooled over all segments.
- **Dead time removed:** the share of non-play time not played. Padding counts against it, and warm-ups count as dead time.

The two trade off directly through the padding, so they must be read together. Each video's labels cover seconds 300 to 900 of a 20-minute clip.

5.3 Cross-validation

The 42 labeled videos are sorted by lighting group (day, dusk, indoor, dim indoor, night) and dealt round-robin into 4 folds of 11, 11, 10 and 10 videos. Each fold's model trains on all other videos and predicts its held-out videos.

The decode and playback settings, the short-segment weight, the resize method and a model-averaging test were all chosen on the same out-of-fold predictions the headline is computed from. To estimate the effect of that selection, the decode was also chosen on three folds and scored on the fourth (nested selection).

5.4 Results

	Points kept whole	Dead time removed
Original selection (decode chosen on all folds)	94.2% (1,272 / 1,351)	50.4% mean of videos
Same, dead time weighted by time		58.7%
Nested selection (chosen on 3 folds, scored on the 4th)	**91.9%**	**52.4%**
Without 5 edited, near-continuous videos (978 points)	92.2%	57.2%
95% bootstrap interval over videos (original)	91.1% to 96.4%	42% to 58%

Table 4: Point segmentation, 4-fold cross-validation over 42 videos and 1,351 labeled serve segments.

Selecting the decode without the scored fold costs about 2 points of points kept whole (Table 4). The 91.9% result measures sensitivity to decode selection. It is not fully nested model cross-validation: selecting-fold predictions come from models trained using the scored fold, and architecture and other design choices remain selected on the full experiment. A sealed match-level test set or fully nested retraining is needed for a cleaner generalization estimate.

Five videos are edited, near-continuous play, and they flatter the headline. All their points are kept and almost none of their dead time is removed. One of them scores 100% kept whole although the model found only 3% of its play, simply because there is almost nothing to cut. Per-fold results range from 90.6% to 96.6% kept whole.

Seed	Kept whole, shipped decode	Dead removed, mean / time	Kept whole, nested	Dead removed, nested
0	94.2%	50.4% / 58.7%	91.9%	52.4%
1	94.2%	48.4% / 56.3%	92.4%	50.5%
2	93.9%	48.1% / 56.1%	92.0%	51.2%

Table 5: All four folds retrained from scratch with three seeds. The shipped decode was chosen on seed 0′s predictions only.

Retraining all four folds with two more seeds changes kept whole by about half a point and dead time removed by about 2 points (Table 5). That is smaller than the selection effect and well inside the bootstrap interval over videos.

Change	Kept whole	Note
None (shipped configuration)	94.2%	original selection
No extra weight on short segments	92.9%	worst on indoor and dim indoor video
Averaging two models	92.5%	dead time removed 53.7%
No per-video threshold rescale	90.7%	dead time removed 55.0%
Bilinear instead of area resize	−1.0 point	fold 0 only
Training on heavily compressed copies	86.3% vs 94.7%	folds 0 and 1 only; run abandoned

Table 6: Changes that did not help, measured on the same folds and decode as the shipped configuration unless noted.

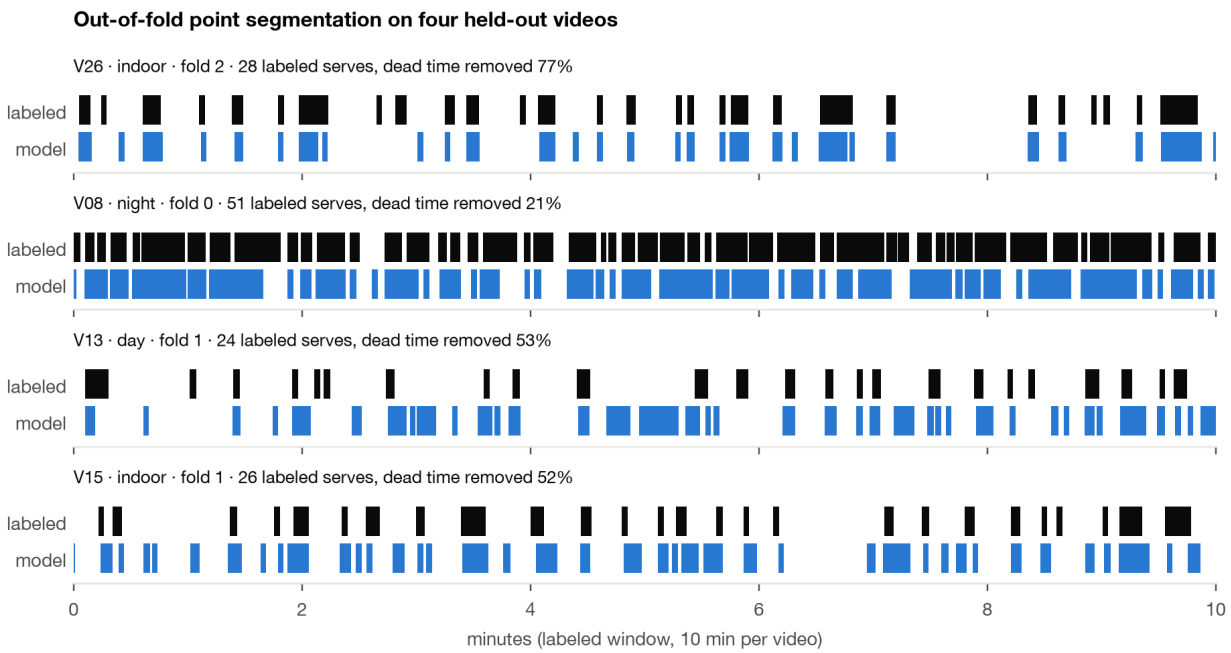


Figure 3: Labeled play and out-of-fold model segments for four held-out videos. The night video (V08) has long rallies and short gaps, so little dead time can be removed without cutting into points.

6 Evaluation methodology

6.1 Leakage in the first splits

The first ball and court training code split individual samples at random. Ball samples are 5-frame windows that slide one frame at a time, so neighboring windows share four of five frames. Court samples are about 18 frames per fixed camera. According to the measurements recorded when the problem was found:

- 99.1% of validation ball windows had a training window one frame away.
- 99.8% of validation court frames came from a source video also in training.

The validation accuracies at the time (0.91 ball detection, 0.99 court keypoints) were therefore mostly memorization. The fix assigns whole groups to validation by a hashed rank: clips for the ball, source videos for the court, and videos for the point segmenter. Held-out ball detection accuracy dropped to 0.812, though the scoring tolerance also changed from 6 or 8 px to 4 px, so the two numbers are not directly comparable. The raw logs behind the overlap figures were not kept, so they are reported as documented, not reproduced.

6.2 What still limits the evaluation

- **Small, correlated test sets.** Frames within a video are strongly correlated. Point-segmentation intervals resample videos; the descriptive ball Wilson interval and McNemar test are frame-level and do not account for that correlation.
- **Selection on evaluation data.** The ball pooling grids, the court checkpoint choice and the point decode were all tuned on the data they are reported on. Where possible, a nested or untouched estimate is reported next to the original.

- **Remaining leakage.** Videos from the same channel or event can sit in different point folds, because no channel metadata exists to group by. The ball split groups by clip, not match. The broadcast court set overlaps its training split.
- **Labels.** Benchmark annotation combines automatically generated drafts, model candidate positions, and final author review. There is no second independent human annotator or agreement measurement. The ball review sheets showed the evaluated model's own candidates (7 accepted positions are its candidates, 116 the teacher's), which can bias labels toward positions the models find.
- **Few seeds.** The original point runs used one seed. Two repeats were added for this report.

7 A new match

As an end-to-end check on footage none of the models had seen, all three ran on my own recording of a college doubles match: 20 minutes at 1280×720 and 60 fps, filmed from behind the near baseline. The author reports permission from everyone shown for use of the demonstration footage. The recording's audio track is silent, so the segmenter, trained with audio, ran on vision features alone.

The match was labeled from its frames alone, without access to any model output: 42 serve segments (26 points, 15 faults, 1 let). A second review pass changed one end time.

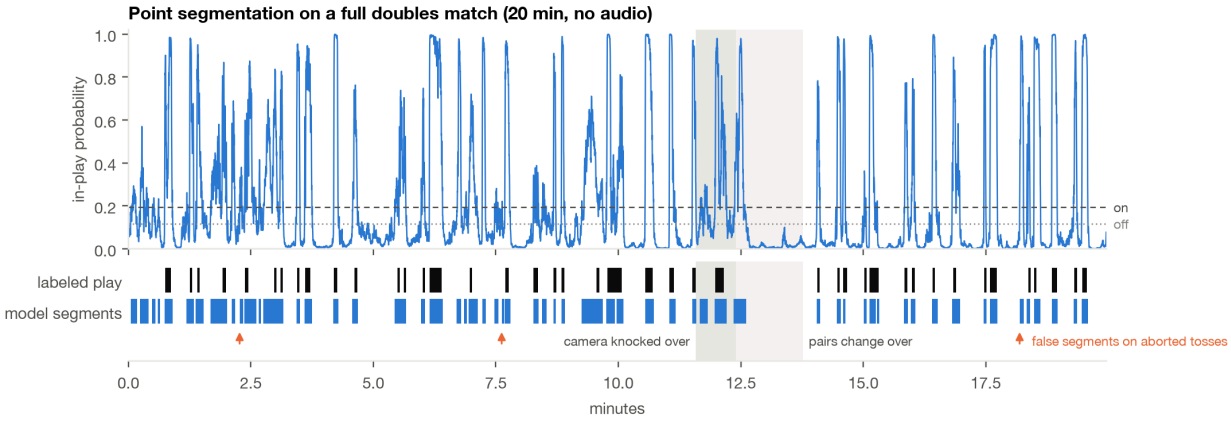


Figure 4: Point segmentation on the full match. Top: in-play probability with the effective decode thresholds after the per-video rescale. Bottom: labeled play and decoded segments.

All 42 serves were kept whole, and 54.6% of dead time was removed (Figure 4). The court locked on 96.0% of half-second samples, with a median reprojection error of 9.5 cm. Of 59 predicted segments, 17 contain no labeled play (92 s in total):

- **Aborted serves:** several fire where the server tossed, stopped and caught the ball. The model sees a serve starting and has no way to know it was called off.
- **Setup:** others come from the pre-match setup.
- **The knocked-over phone:** the court gate correctly rejected the tilted view and paused ball tracking for 47.5 s, but the segmenter kept running and produced false segments while players walked.

The 40 seconds of empty court during the pair changeover produced none. One video cannot estimate a rate; this is a demonstration, not a benchmark.

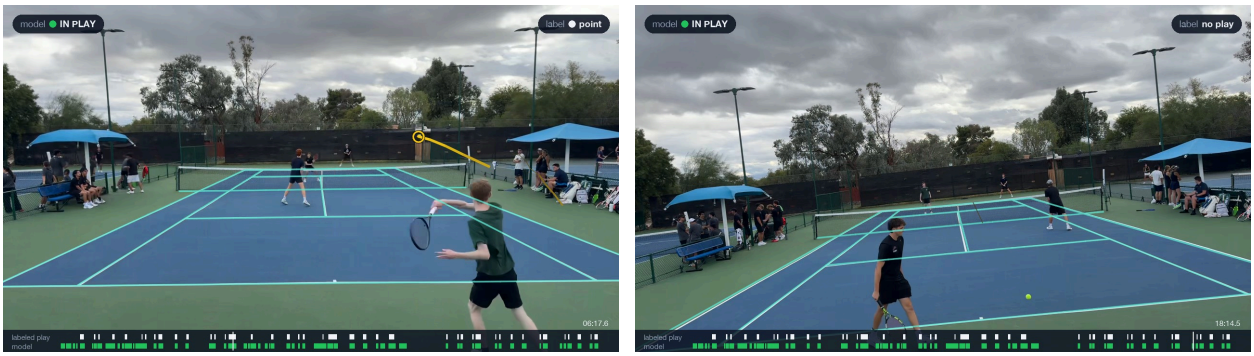


Figure 5: Overlay of the shipped models on the match. Left: a rally with the tracked ball (yellow), the projected court model (teal), the segmenter's decision and the label. Right: a false in-play segment on an aborted serve.

8 Discussion and next experiments

The evidence supports an applied research contribution in robust perception and evaluation. It does not validate autonomous officiating: court lock is a geometric acceptance signal, and an inlier residual is not independently measured position error. Bounce detection, score inference, and player statistics remain outside the evaluated scope. Native mobile integrations exist, but this report does not provide a reproducible device latency, memory, or energy benchmark.

The next study should freeze a match-level test set before model selection, group related cameras and events, and obtain independent annotations without model candidates. A fully nested training protocol would isolate both architecture and decode selection. Audio-only, visual-only, and combined models would test the value of each modality. Phone benchmarks should record hardware, runtime, precision, sustained throughput, and thermal conditions. These are proposed experiments, not completed results.

9 Author contribution

Jacob Beal is the sole author and developer. He conceived and directed the project, including implementation, annotation review, training, evaluation, and integration into MyBreakPoint. This work was conducted as independent product research at Break Point Technologies, LLC.

10 Reproducibility and availability

The repository contains:

- the model, tracker, geometry, splitting, training and evaluation code;
- unit tests on synthetic data;
- result tables with anonymized video identifiers;
- the figures and demo media in this report.

The following are withheld:

- **Footage:** public YouTube videos without redistribution rights, with identifiable players.
- **Labels:** tied to that footage.
- **Trained weights and tuned operating values** (tracker thresholds and gates, decode thresholds and durations, court-gate limits, lighting calibration): the models ship in a commercial app.

The published numbers can be checked against the code and tables, but not rerun from scratch. Two runs can't be repeated at all, because their data and logs were not kept: the training behind the evaluated ball weights and the first court model.

Several statements in the original project notes had no saved output and are excluded from this report: 60 fps robustness, the distilled student's accuracy on the labels, fp32 versus fp16 accuracy, and the lighting-simulation calibration measurements.

References

[1] Y.-C. Huang, I.-N. Liao, C.-H. Chen, T.-U. İk, and W.-C. Peng, "TrackNet: A Deep Learning Network for Tracking High-speed and Tiny Objects in Sports Applications," in *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2019. [Online]. Available: https://arxiv.org/abs/1907.03698

[2] N.-E. Sun *et al.*, "TrackNetV2: Efficient Shuttlecock Tracking Network," in *International Conference on Pervasive Artificial Intelligence (ICPAI)*, 2020. doi: 10.1109/ICPAI51961.2020.00023.

[3] S. Tarashima, M. A. Haq, Y. Wang, and N. Tagawa, "Widely Applicable Strong Baseline for Sports Ball Detection and Tracking," in *British Machine Vision Conference (BMVC)*, 2023. [Online]. Available: https://arxiv.org/abs/2311.05237

[4] A. Raj, L. Wang, and T. Gedeon, "TrackNetV4: Enhancing Fast Sports Object Tracking with Motion Attention Maps," *arXiv preprint arXiv:2409.14543*, 2024, [Online]. Available: https://arxiv.org/abs/2409.14543

[5] yastrebksv, "TennisCourtDetector: Deep learning network for detecting tennis court." Accessed: Nov. 18, 2026. [Online]. Available: https://github.com/yastrebksv/TennisCourtDetector

[6] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal Convolutional Networks for Action Segmentation and Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. doi: 10.1109/CVPR.2017.113.

Appendix: per-video results

Video	Visible	Within 8 px	Within 3 px	No output	False alarms
V02	41	38 (92.7%)	31	2	3 / 7
V11	31	26 (83.9%)	22	4	4 / 18
V14	29	26 (89.7%)	26	2	5 / 16
V20	36	33 (91.7%)	33	3	2 / 14
V22	34	27 (79.4%)	27	5	3 / 15
V26	42	41 (97.6%)	38	1	1 / 7
V35	35	27 (77.1%)	27	6	2 / 11

Table 7: Ball tracking per test video (shipped model and tracker). False alarms are counted over each video's no-ball frames.

Video	Light	Fold	Serves	Kept	Dead
V01	day	0	76	100.0%	0.0%
V02	day	0	22	100.0%	58.5%
V03	day	0	34	97.1%	67.0%
V04	indoor	0	21	100.0%	65.1%
V05	indoor	0	17	100.0%	82.3%
V06	indoor	0	50	100.0%	28.1%
V07	dim	0	0		82.8%
V08	night	0	51	84.3%	21.5%
V09	night	0	15	93.3%	88.8%
V10	night	0	61	96.7%	15.5%
V11	night	0	39	97.4%	34.0%
V12	day	1	27	92.6%	47.7%
V13	day	1	24	66.7%	53.3%
V14	day	1	20	100.0%	72.6%
V15	indoor	1	26	73.1%	52.2%
V16	indoor	1	45	95.6%	17.8%
V17	indoor	1	18	100.0%	70.8%
V18	dim	1	20	85.0%	76.7%
V19	night	1	21	100.0%	68.9%
V20	night	1	26	100.0%	29.4%
V21	night	1	27	100.0%	74.3%
V22	night	1	24	100.0%	62.0%
V23	day	2	31	80.7%	52.9%
V24	day	2	87	100.0%	0.0%
V25	dusk	2	17	94.1%	72.6%
V26	indoor	2	28	71.4%	76.7%
V27	indoor	2	37	94.6%	61.7%
V28	dim	2	22	81.8%	69.1%
V29	dim	2	21	90.5%	59.3%
V30	night	2	26	76.9%	77.1%
V31	night	2	71	95.8%	0.0%
V32	night	2	0		19.0%
V33	day	3	73	100.0%	0.0%
V34	day	3	20	100.0%	78.6%
V35	dusk	3	26	96.2%	71.6%
V36	indoor	3	40	100.0%	23.0%
V37	indoor	3	18	83.3%	74.3%
V38	dim	3	41	95.1%	10.1%
V39	dim	3	23	91.3%	75.4%
V40	night	3	24	91.7%	72.4%
V41	night	3	66	100.0%	1.1%
V42	night	3	16	87.5%	54.3%

Table 8: Point segmentation per video, seed 0, shipped decode: labeled serve segments, share kept whole and share of dead time removed ("dim" is dim indoor). Blank entries are videos with no play in the labeled window.