Robust Camera Calibration for Sport Videos using Court Models

Dirk Farin, Susanne Krabbe,
Peter H. N. de With, Wolfgang Effelsberg

Contact address:
Dirk Farin
Eindhoven University of Technology (TU/e)
Embedded Systems Institute
5600 MB, Eindhoven, The Netherlands
d.s.farin@tue.nl

Introduction

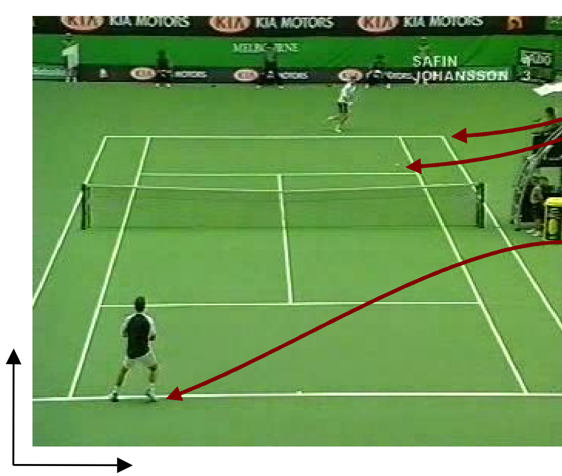
Semantic analysis of sport videos depends on

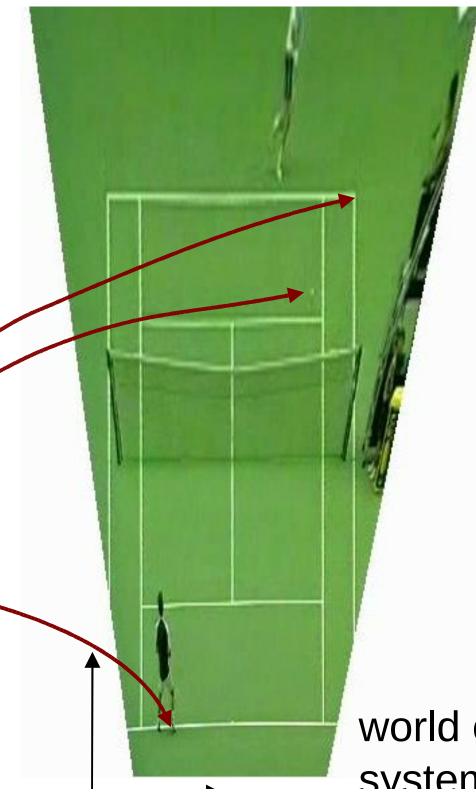
- information about player and ball position,
- position in real-world coordinates, not screen coordinates.

Geometric mapping between screen coordinates and real-world required.

Algorithm design goals:

- robust to occlusion, partial court views,
- generic algorithm, applicable to all court sports,
- support arbitrary camera placement and motion.





Camera Model

- 3D camera set-up and image formation (in homogeneous coordinates):

$$\mathbf{p_i} = \mathbf{H}\mathbf{p}'_\mathbf{i} = \underbrace{\begin{pmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{pmatrix}}_{\text{camera projection}} \underbrace{\begin{pmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \end{pmatrix}}_{\text{camera rotation and placement}} \underbrace{\begin{pmatrix} x' \\ y' \\ z' = 0 \\ 1 \end{pmatrix}}_{\text{world coordinate}}$$

assume that court ground is at z=0

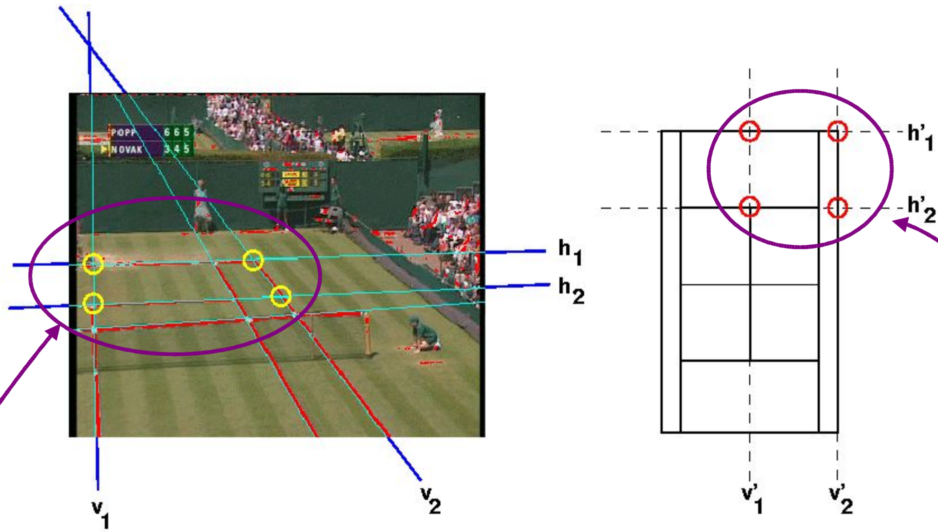
- multiplying this out leads to 8-parameter perspective motion model:

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}}, \quad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}}$$

- 8 parameters ➙ 8 equations required to solve for parameters
- ➙ 4 point correspondences required (each gives 2 equations)

Principle of Calibration Algorithm

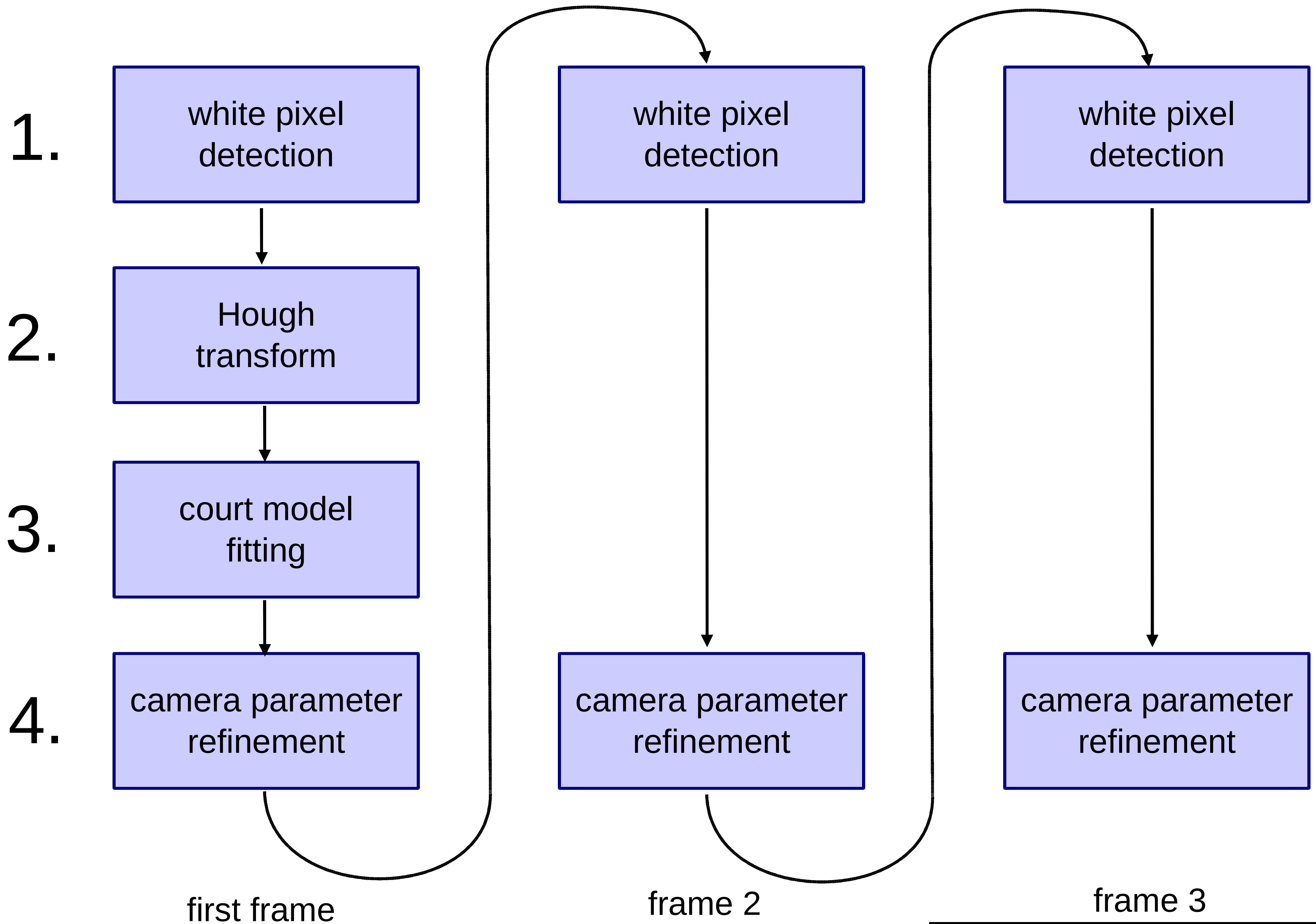
- Select (arbitrarily) two horizontal and two vertical court lines.
- Determine corresponding court lines / points in court model.
- Intersection gives four intersection points.



- Use point correspondences to solve for camera parameters:

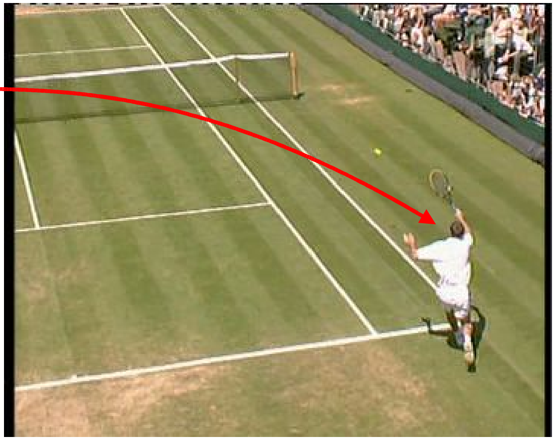
$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & x_1x'_1 & y_1x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & x_1y'_1 & y_1y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & x_2x'_2 & y_2x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & x_2y'_2 & y_2y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & x_3x'_3 & y_3x'_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & x_3y'_3 & y_3y'_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & x_4x'_4 & y_4x'_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & x_4y'_4 & y_4y'_4 \end{pmatrix} \boxed{\begin{pmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \end{pmatrix}} = \begin{pmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{pmatrix}$$

Flowgraph of Camera Calibration Algorithm



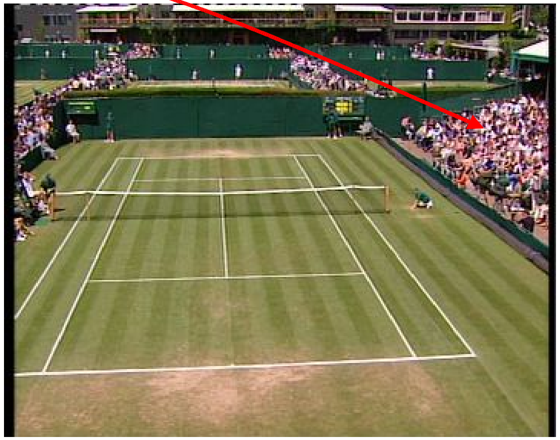
White (Court-Line) Pixel Detection (1/3)

Detect white pixels belonging to court-lines.



Challenges:

- Players often dressed in white
  (large white regions)
- White colors in the audience, stadium
  (fine textured areas)



Each pixel must pass several tests:

- above fixed brightness threshold ?
  - exclude dark regions
- brighter than neighborhood ?
  - exclude homogeneous areas
- linear structure in neighborhood ?
  - exclude textured areas

Tests have increasing complexity.

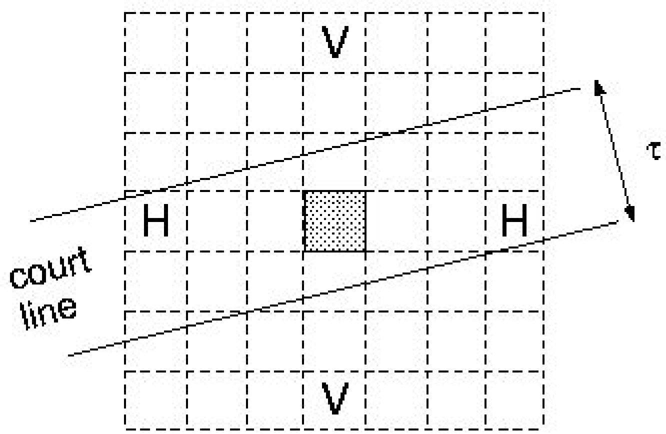
White (Court-Line) Pixel Detection (2/3)

Flat area exclusion rule (test 2):

- pixel candidate must be brighter than

  both pixels marked 'H', or

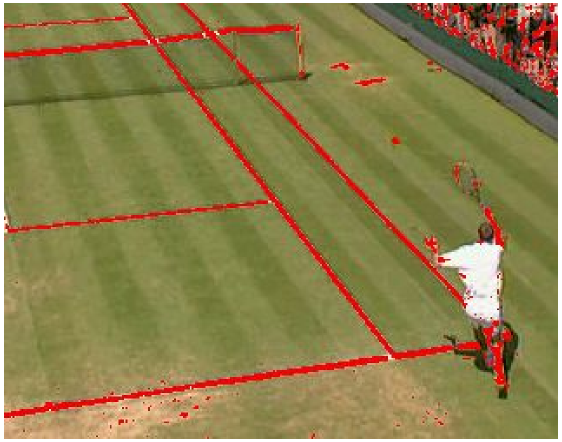
  both pixels marked 'V'.



- Exclusion of homogeneous white areas

  with very low computational cost.

input frame



remaining pixel candidates

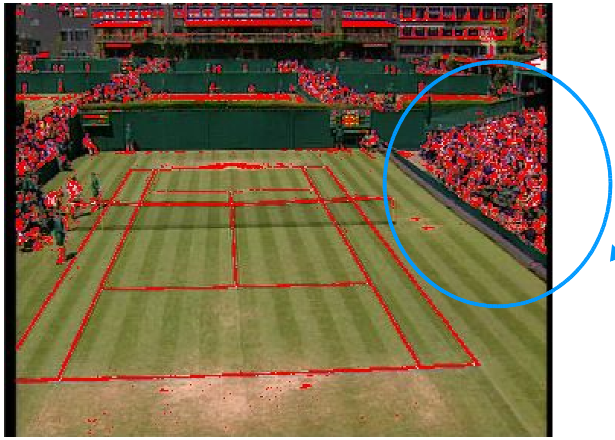
White (Court-Line) Pixel Detection (3/3)

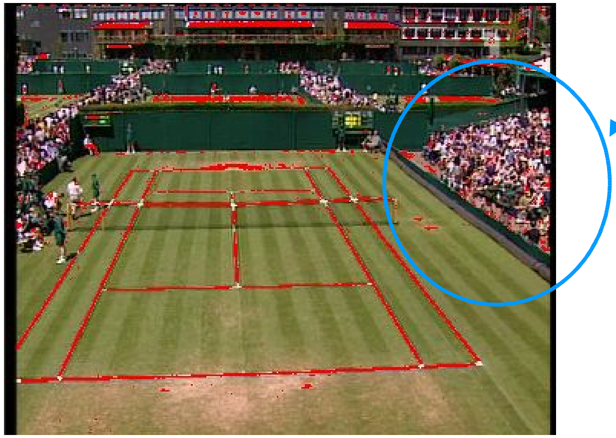
Textured area exclusion rule (test 3):

- compute structure matrix within the pixel neighborhood:

$$\mathbf{S} = \begin{pmatrix} J_{xx} & J_{xy} \\ J_{xy} & J_{yy} \end{pmatrix} = \sum_{x=p_x-b}^{p_x+b} \sum_{y=p_y-b}^{p_y+b} \nabla g(x,y) \cdot (\nabla g(x,y))^T$$

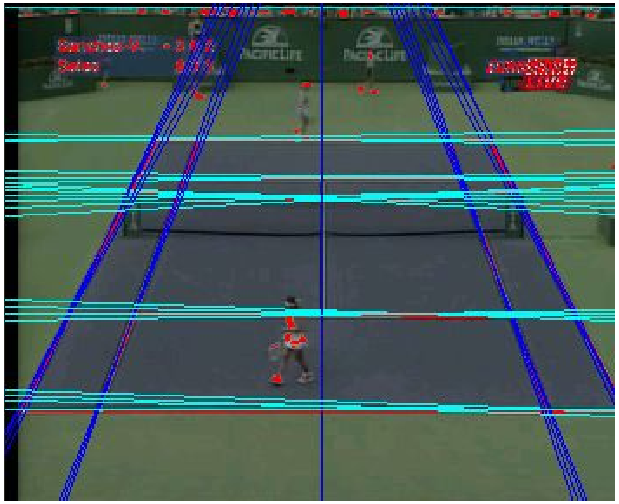
- structure can be classified by evaluating the magnitude of the two eigenvalues.
  - $\lambda_1 >> \lambda_2 \rightarrow$ linear structure
- complex test, but has to be applied only to remaining candidate pixels



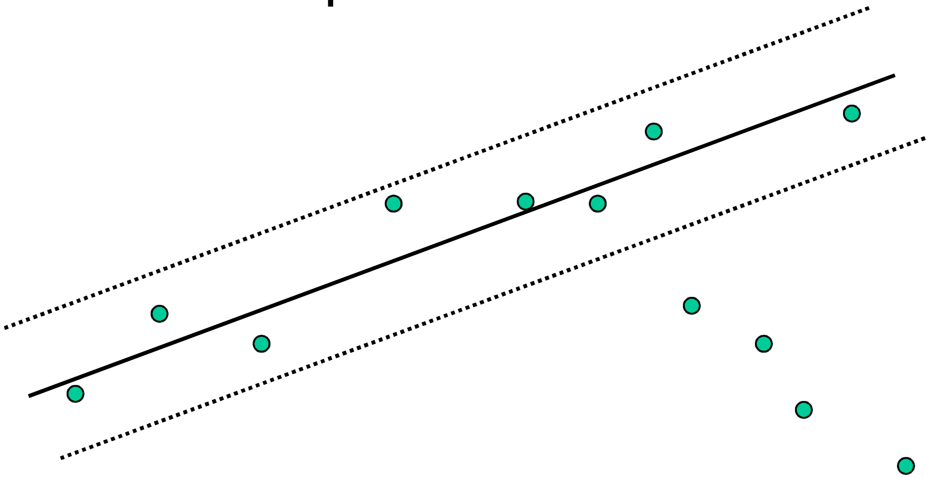


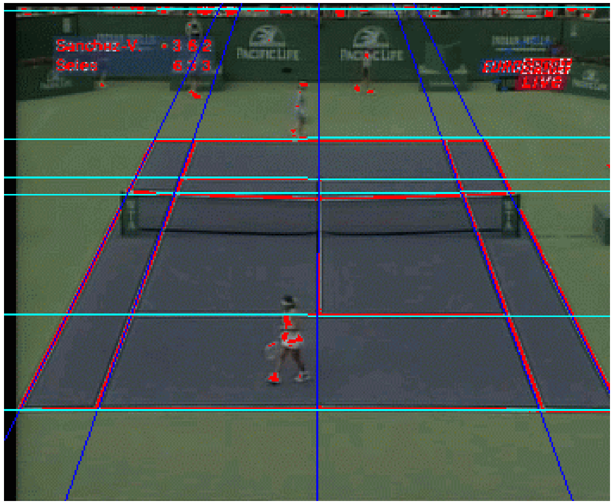
Court-Line Detection

- Apply Hough-transform on *white* pixels to find line candidates.
  - Usually, too many line candidates are found.



- Apply robust regression algorithm (*LTS*) to refine line parameters.





- Remove duplicate lines if their parameters are (almost) equal.

Determining Court-Line Correspondences

detected lines

court-model lines

classify detected lines as horizontal or vertical

fixed court model lines, divided into h./v. lines

iterate through all hor./vert. line pairs

iterate through all hor./vert. line pairs

calculate line intersection points

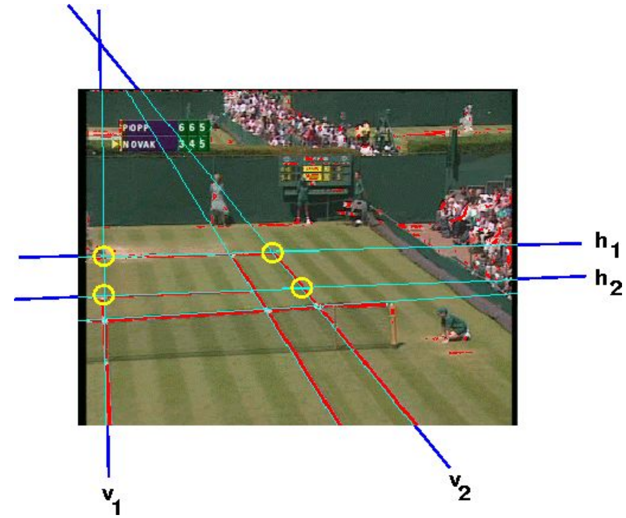
calculate line intersection points

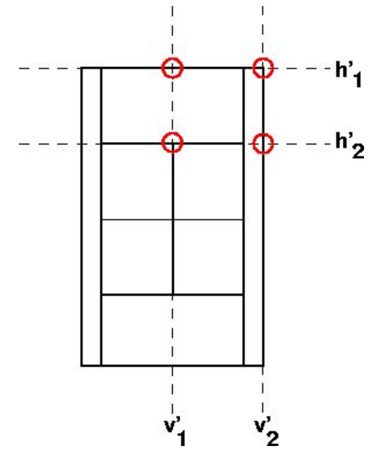
build equation system from intersection points data and solve for camera parameters

exclude obviously incorrect parameter sets (see two slides later)

measure model support (see next slide)

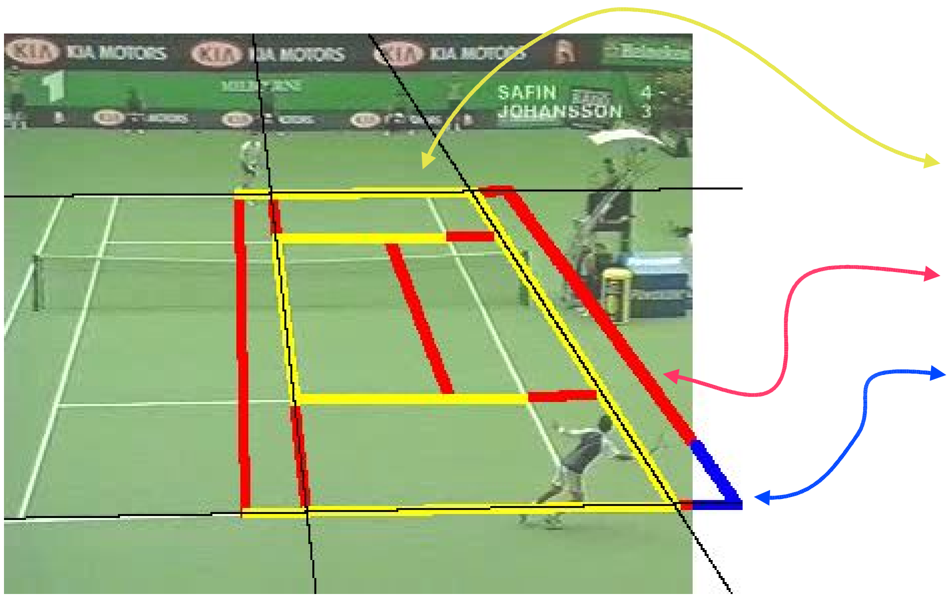
select parameter set with largest support





Selecting the Best Parameter Set

- Use obtained camera parameters to project model back onto image.
- Count court-line pixels that are covered by the model:

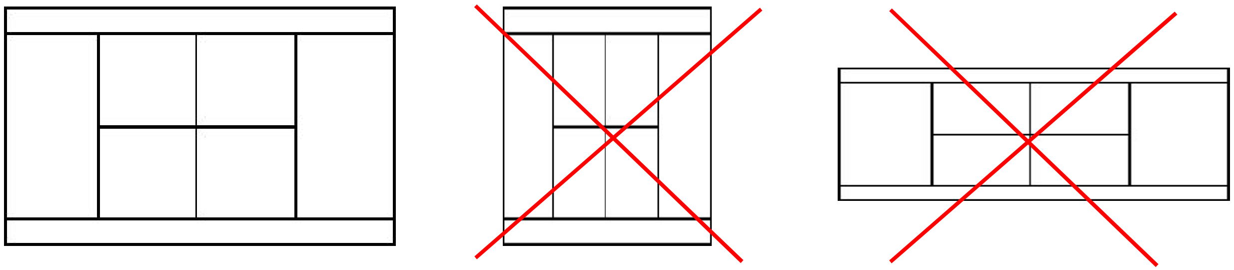


	weight
on court line	1
not on court line	-1/2
outside of image	0

- The parameter set which gives the highest score is selected.
- But: before measuring score, perform parameter sanity check:
  - avoids computation of score if parameters are obviously wrong,
  - increases robustness by excluding impossible parameter values.

Fast Parameter Set Exclusion Rule (1/2)

- Camera model allows non-isotropic scaling, which is impossible in the real world.



- Non-isotropic scaling in our camera model:

$$\mathbf{p_i} = \mathbf{H}\mathbf{p}'_\mathbf{i} = \underbrace{\begin{pmatrix} f & 0 & o_x \\ 0 & f & o_y \\ 0 & 0 & 1 \end{pmatrix}}_{\substack{\text{internal camera} \\ \text{parameters}}} \underbrace{\begin{pmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \end{pmatrix}}_{\substack{\text{camera rotation,} \\ \text{translation}}} \boxed{\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\substack{\text{non-isotropic} \\ \text{scaling}}}} \begin{pmatrix} x' \\ y' \\ z' = 0 \\ 1 \end{pmatrix}$$

β should always be equal to 1

Fast Parameter Set Exclusion Rule (2/2)

- Decompose transformation matrix H to obtain β.
- Assume that principal point is known.

$$\begin{pmatrix}1 & 0 & -o_x\\ 0 & 1 & -o_y\\ 0 & 0 & 1\end{pmatrix}\mathbf{H} = \mathbf{H}' = \begin{pmatrix}f & 0 & 0\\ 0 & f & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}r_{00} & r_{01} & t_x\\ r_{10} & r_{11} & t_y\\ r_{20} & r_{21} & t_z\end{pmatrix}\begin{pmatrix}1 & 0 & 0\\ 0 & \beta & 0\\ 0 & 0 & 1\end{pmatrix} = \begin{pmatrix}fr_{00} & \beta f r_{01} & ft_x\\ fr_{10} & \beta f r_{11} & ft_y\\ r_{20} & \beta r_{21} & t_z\end{pmatrix}$$

these are our camera parameters (except a scaling factor)

- We know that { r_i } is orthonormal (rotation matrix).

$$\frac{h'^2_{00}}{f^2} + \frac{h'^2_{10}}{f^2} + h'^2_{20} = \frac{h'^2_{01}}{\beta^2 f^2} + \frac{h'^2_{11}}{\beta^2 f^2} + \frac{h'^2_{21}}{\beta^2}$$

$$\frac{h'_{00}h'_{01}}{\beta f^2} + \frac{h'_{10}h'_{11}}{\beta f^2} + \frac{h'_{20}h'_{21}}{\beta} = 0$$

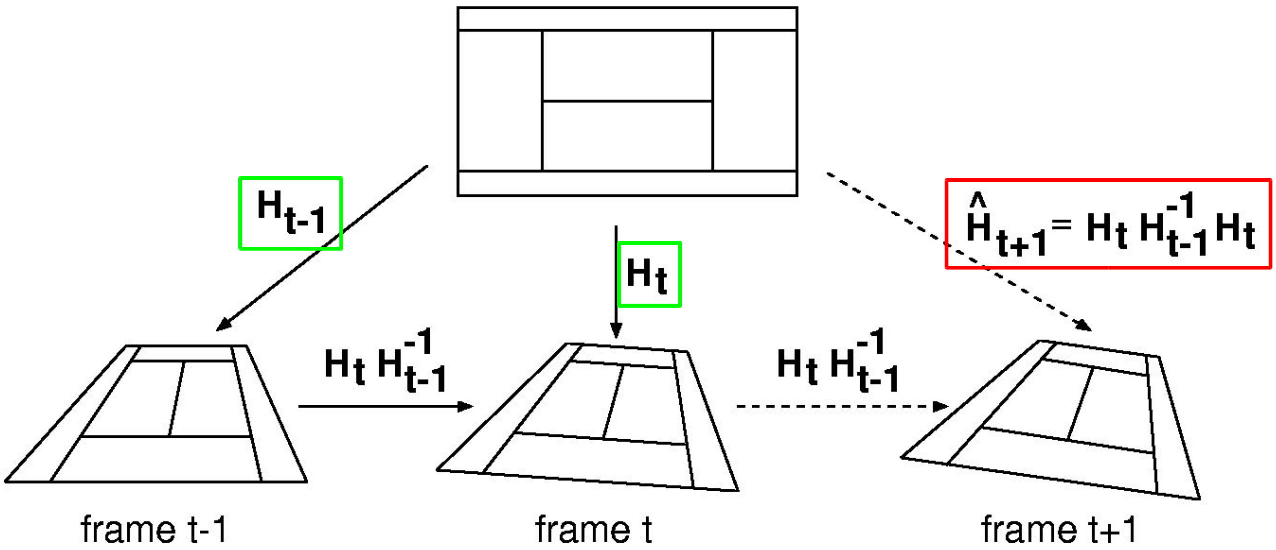
- Hence, we obtain as:

$$\beta^2 = \frac{h'^2_{01} + h'^2_{11} + f^2 h'^2_{21}}{h'^2_{00} + h'^2_{10} + f^2 h'^2_{20}} \quad ; \quad f^2 = -\frac{h'_{00}h'_{01} + h'_{10}h'_{11}}{h'_{20}h'_{21}}$$

- Consider a parameter set as invalid if β<0.5 or β>2.0

Model Tracking

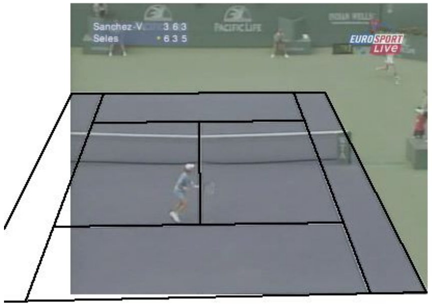
- Algorithm so far operated on single frames.
- When applied on a video-sequence, previous parameters sets can be used to compute an initial estimate.

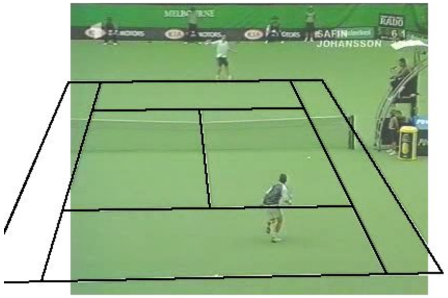


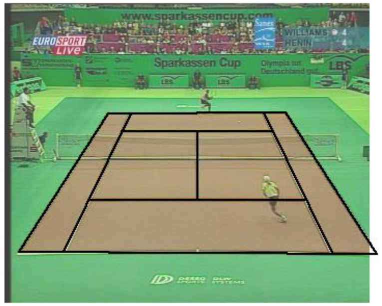
- Optimize camera parameters by minimizing distance between model lines and court-line pixels.
- Non-linear optimization problem ➙ use gradient-descent algorithm.

Results (*Tennis*; 1/2)

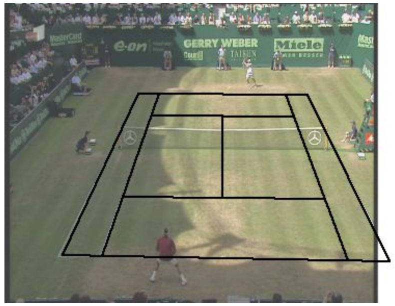
- different court types:



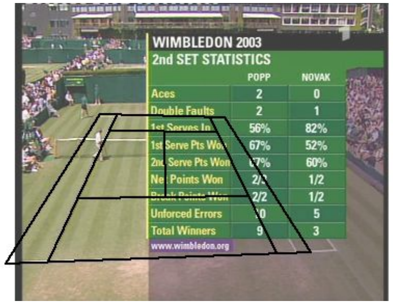




- strong shadow:

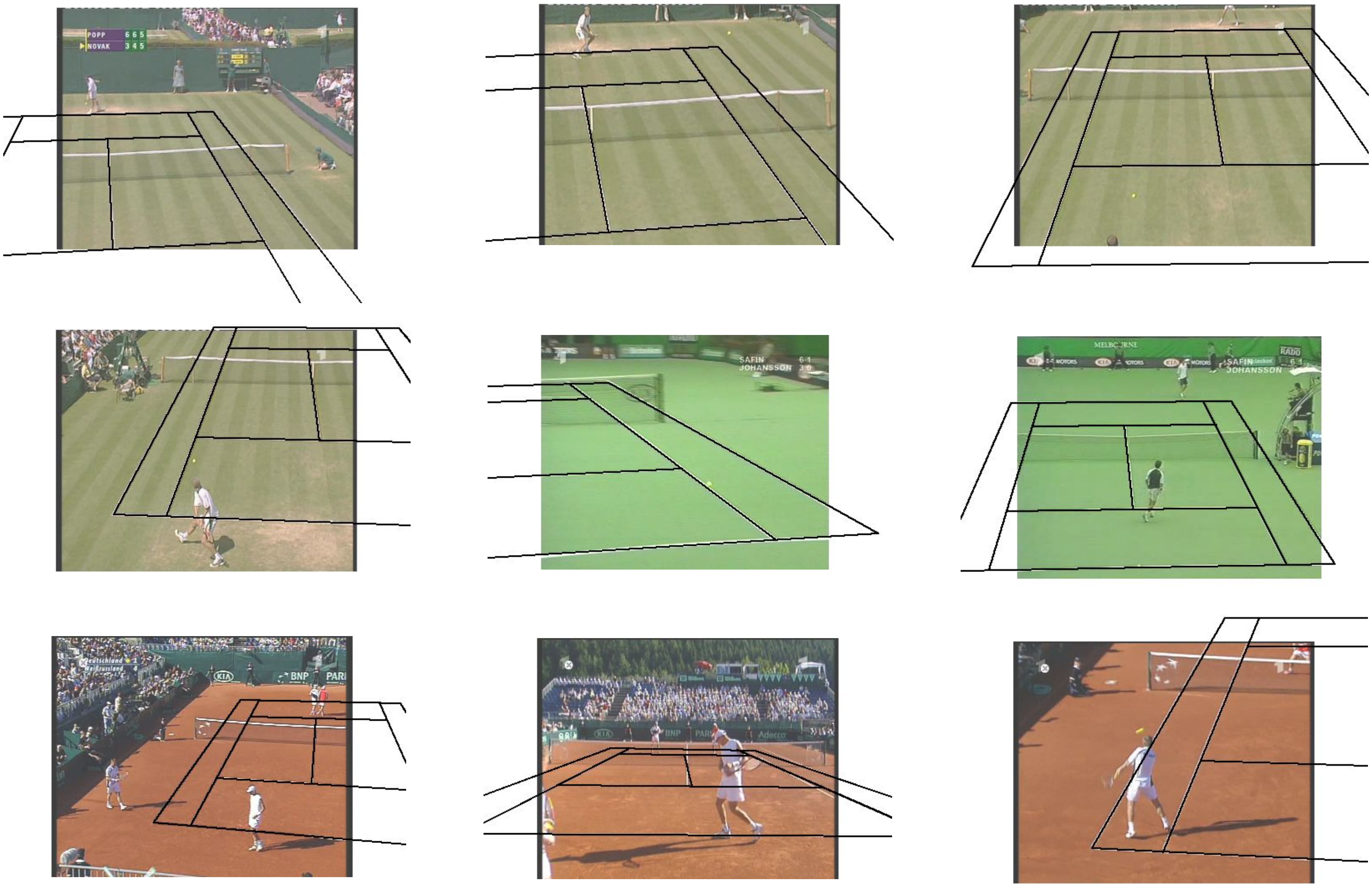


- very large occlusion:

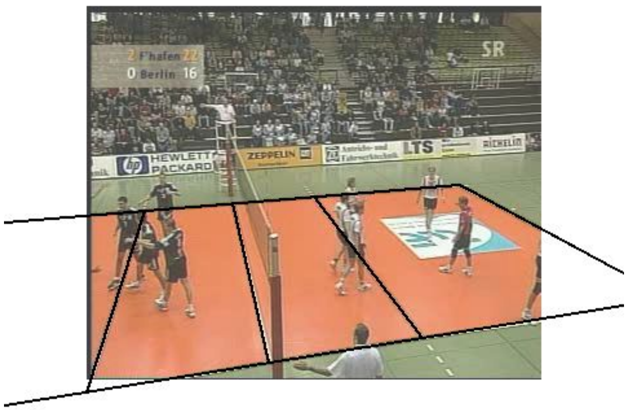


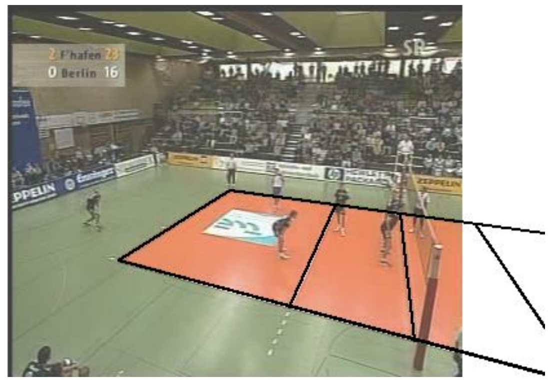
All results were obtained without any parameter adjustment.

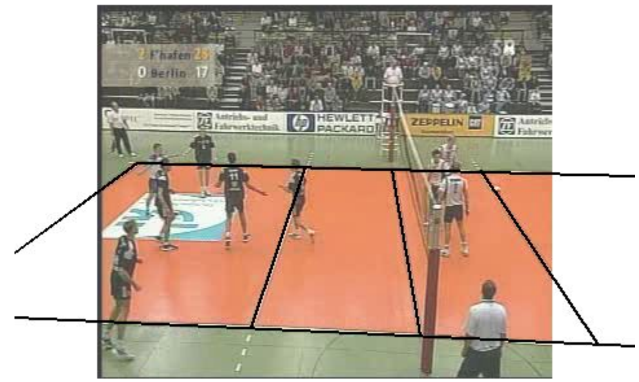
Results (*Tennis*; 2/2)

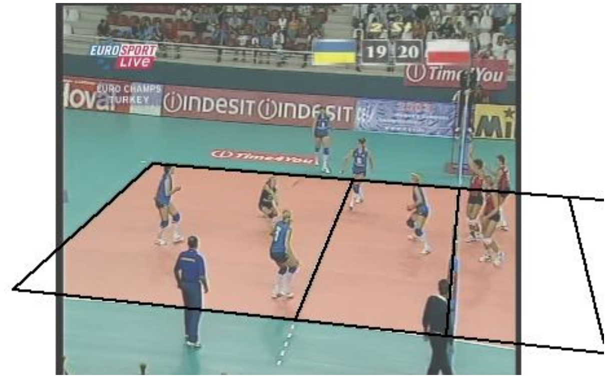


Results (*Volleyball*)

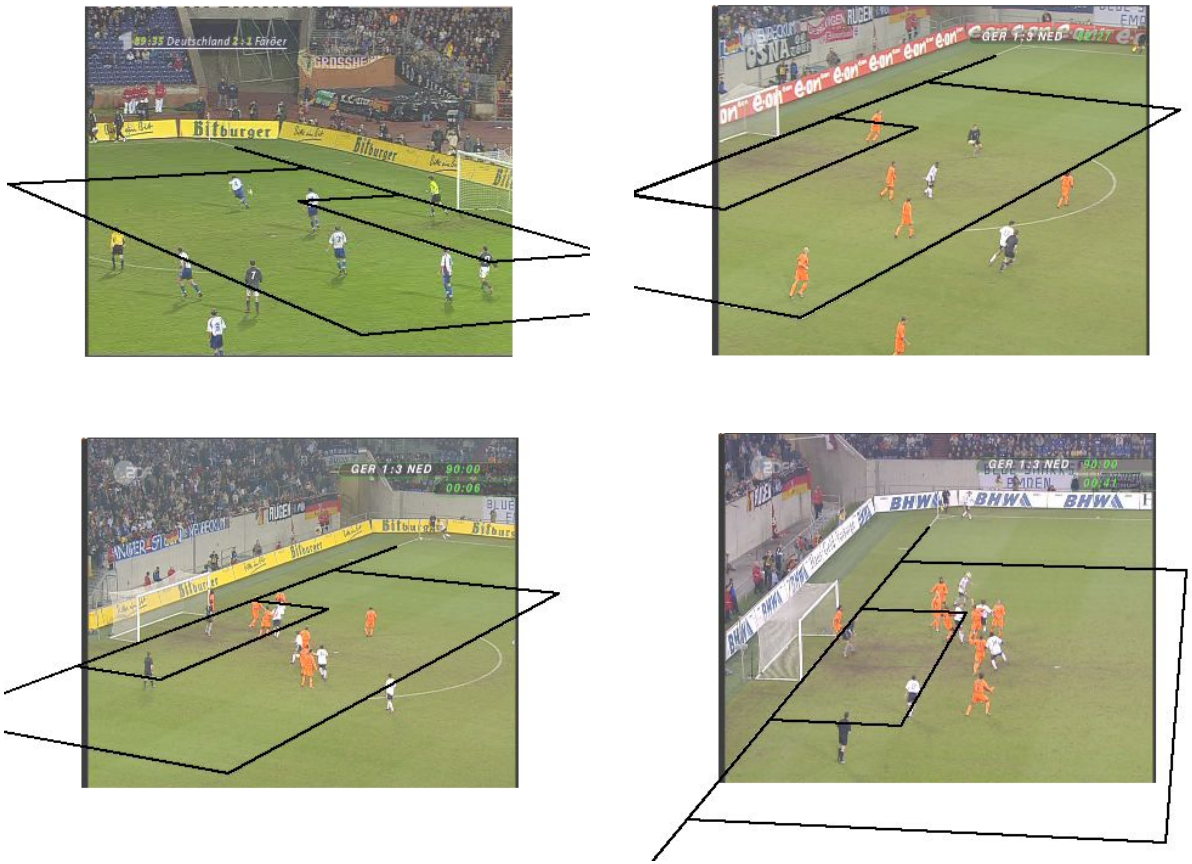








Results (*Soccer*)



- Only goal-area scenes can be processed.
  - 2 vertical & 2 horizontal lines required,
  - for remaining scenes, more restriced camera model is required, or some parameters must be known.

Conclusions

- Camera calibration required to convert between screen coordinates and real-world coordinates.

- Generic algorithm, applicable to any court sport:
  - extract line segments of the input image,
  - use model of court-line arrangement,
  - establish correspondences between input lines and model lines.

- Very robust calibration results
  - insensitive to large occlusions,
  - no need for parameter adaptation.

- Possible extensions
  - include curved model primitives (e.g., circle in center of soccer field).