Drawing Road Networks with Focus Regions

Edited from Jan-Henrik Haunert's slides

If you are here

Providence, RI

Fish-Eye Views (here: Yamamoto et al., ACM GIS 2009)

- enlarge areas of interest
- still give the big picture
- ***but*** introduce large distortions

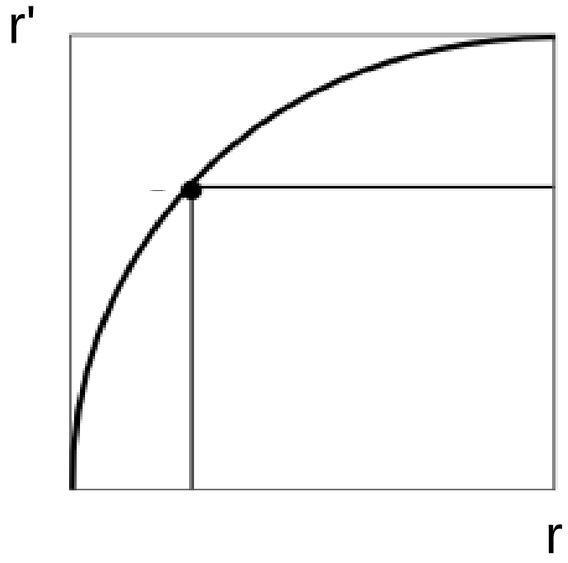
- **Aim:**

  Focus-and-context view with ***minimum distortion***

Providence, RI

Fish-Eye Views (here: Yamamoto et al., ACM GIS 2009)

- *For every pixel (x, y), transform its representation to (θ, r)*
- *Transform r to r' using a spherical function*
  - $r' = r + (1 - sqrt(1 - r^2))$
- *Recalculate (x', y') using*
  - $x' = rcos(\theta)$
  - $y' = rsin(\theta)$



Quadratic Programming

Find $x \in \mathbb{R}^n$ minimizing

- a given ***cost function***

while satisfying

- a given set of ***hard constraints***.

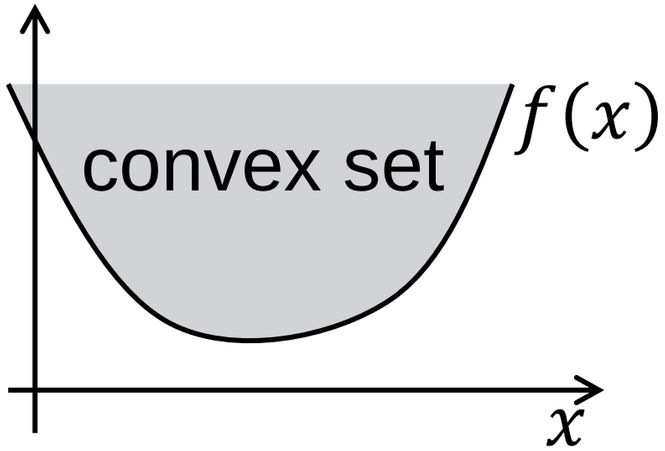
Convex Quadratic Programming

Find $x \in \mathbb{R}^n$ minimizing

- $f(x) = cx + x^T Dx$, where f is a ***convex function***,

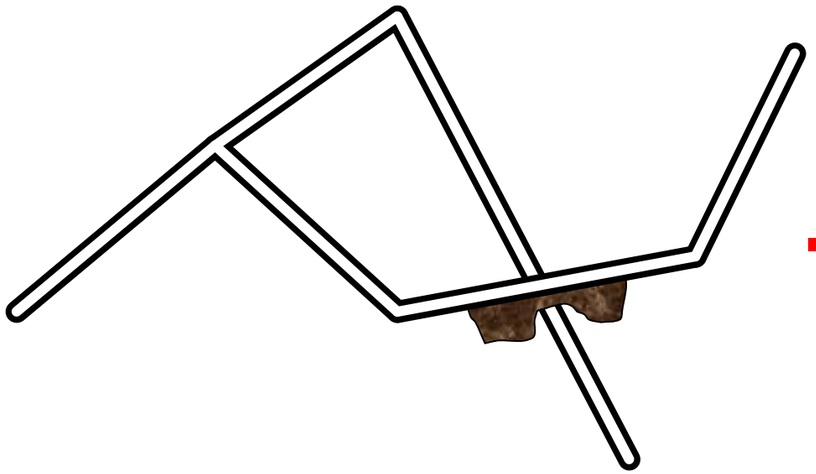
while satisfying

- $Ax \geq b$ and
- $x \geq 0$

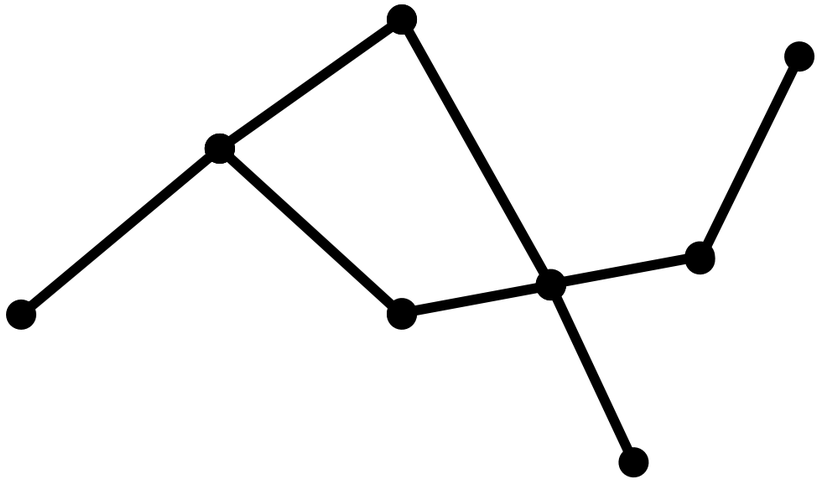


Preliminaries

road network



graph $G = (V, E)$



X_v, Y_v: ***input*** coordinates of node $v \in V$

x_v, y_v: ***output*** coordinates of node $v \in V$

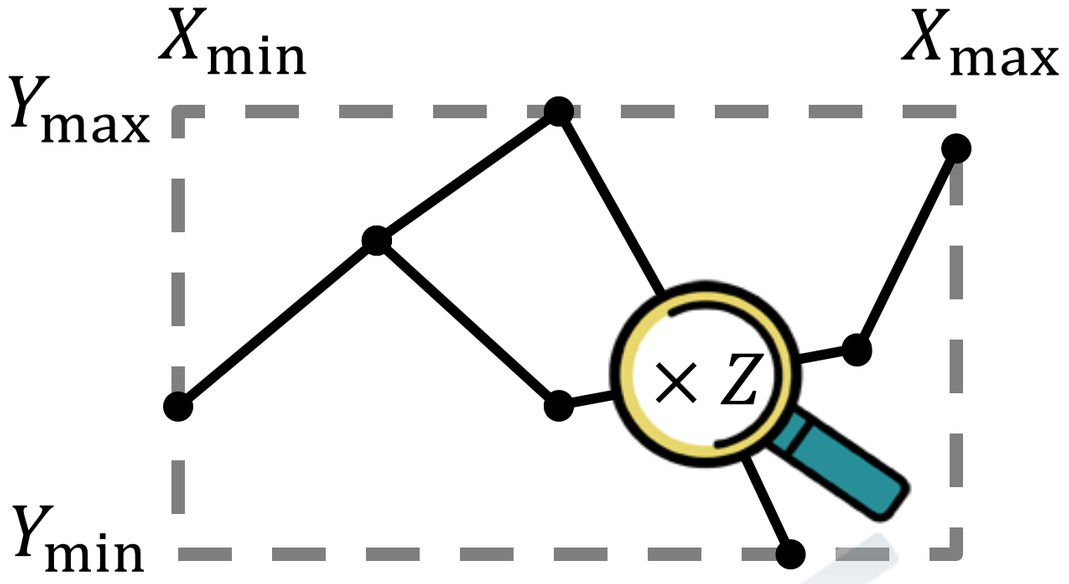
Preliminaries

Output map must fit into given frame!

Constraints:

$$X_{\min} \leq x_v \leq X_{\max}$$
$$Y_{\min} \leq y_v \leq Y_{\max}$$

focus region = subset of node set V

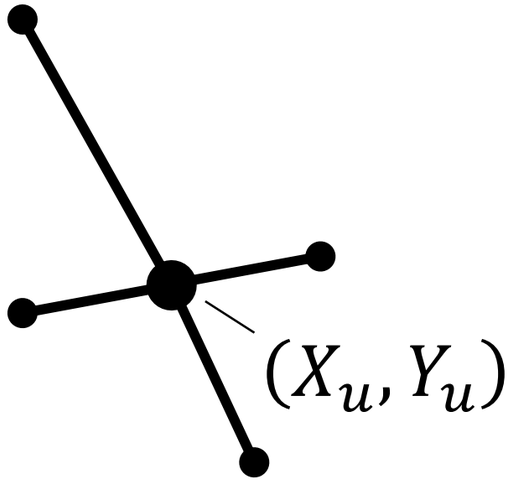


X_v, Y_v: ***input*** coordinates of node $v \in V$

x_v, y_v: ***output*** coordinates of node $v \in V$

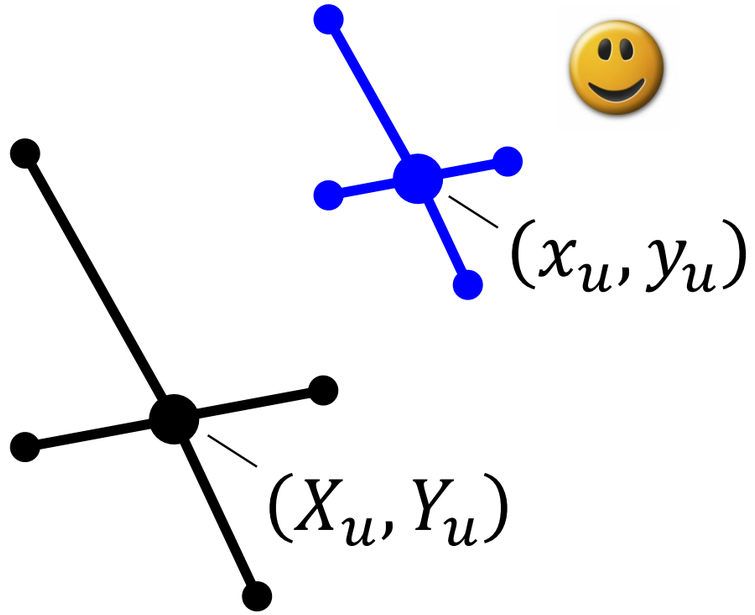
Measuring Distortion

- ***locally***, i.e., for each node $u \in V$ with its neighbors



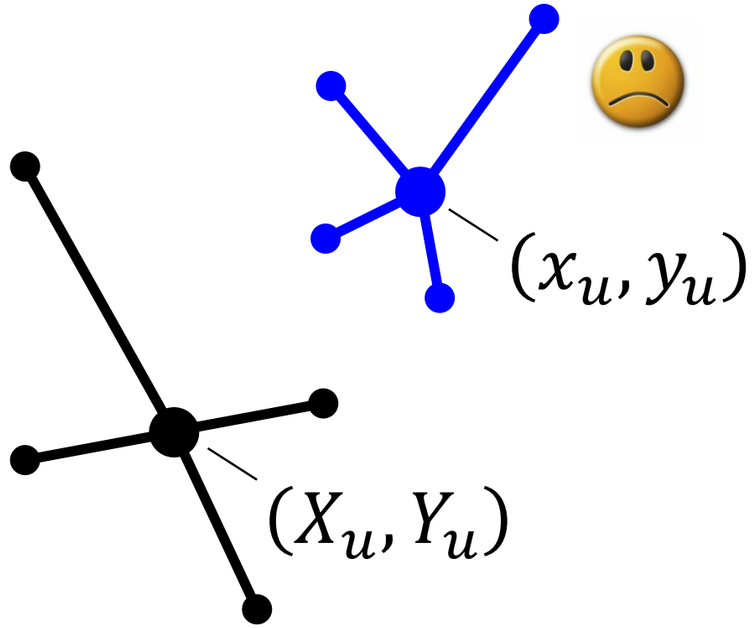
Measuring Distortion

- ***Scaling*** and ***translating***
  is not distorting



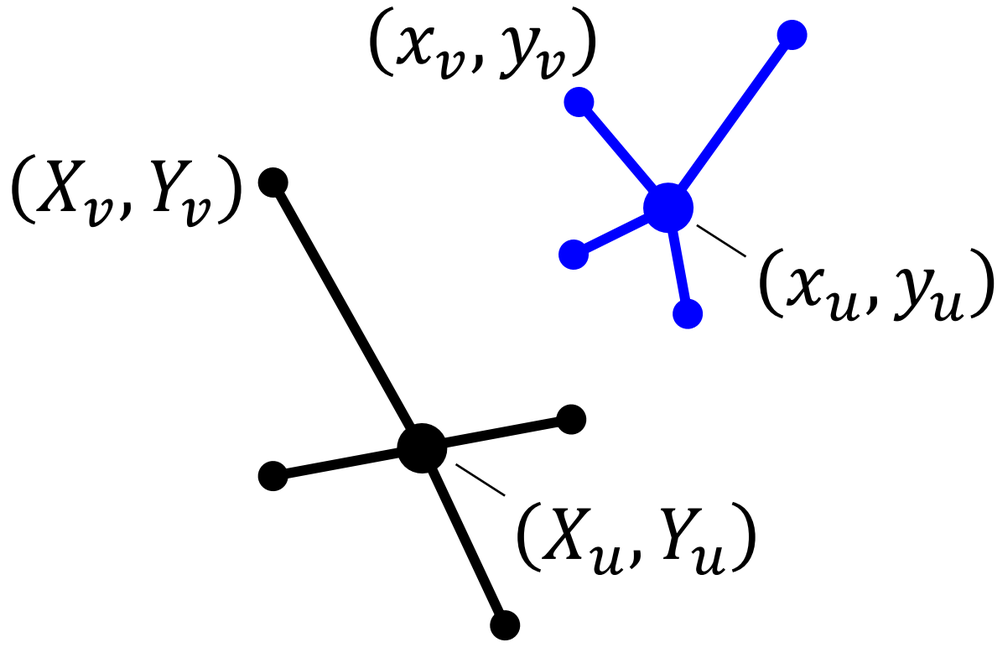
Measuring Distortion

- ***Scaling*** and ***translating*** is not distorting,
- **but every other change is.**



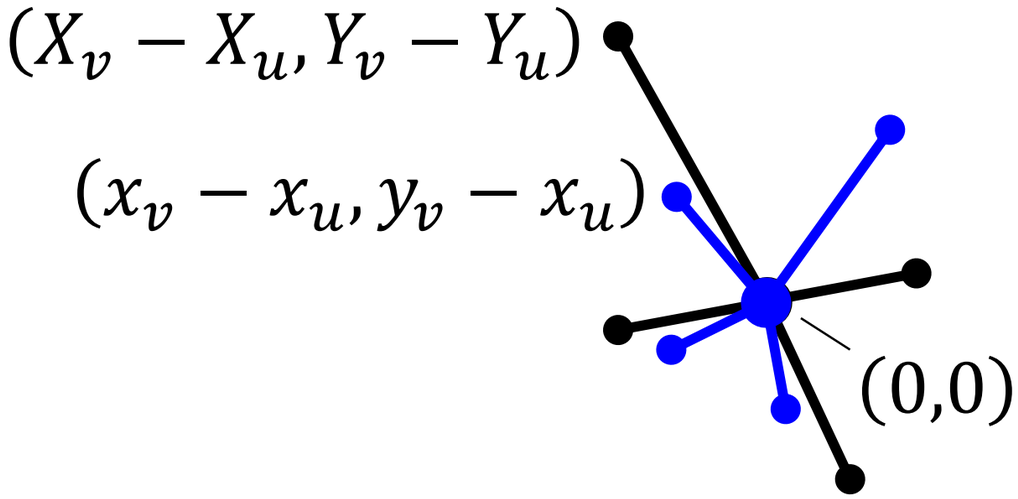
Measuring Distortion

- compare both (sub-)graphs in ***local reference frame***



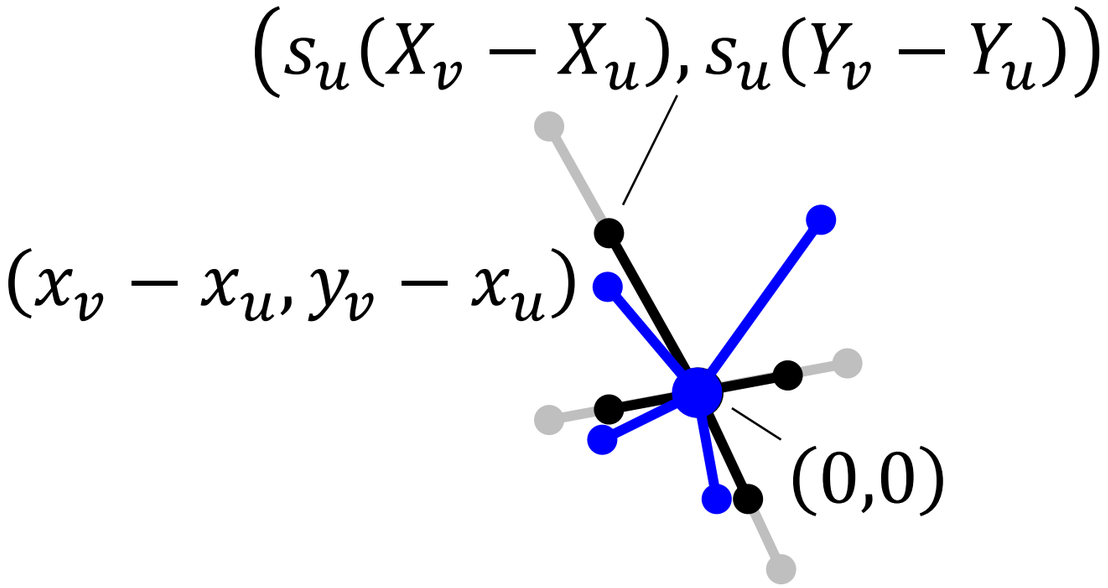
Measuring Distortion

- compare both (sub-)graphs in ***local reference frame***



Measuring Distortion

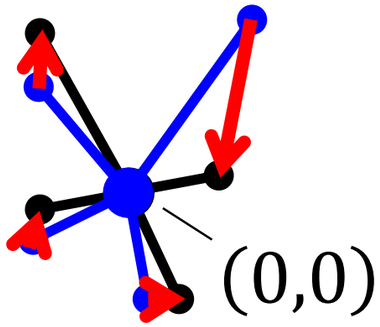
- compare both (sub-)graphs in ***local reference frame***
- apply (unknown) scale factor $s_u \in \mathbb{R}^+$ to compensate scaling



Measuring Distortion

- compare both (sub-)graphs in ***local reference frame***
- apply (unknown) scale factor $s_u \in \mathbb{R}^+$ to compensate scaling
- ***now*** measure ***squared residuals***

$$\begin{pmatrix} s_u(X_v - X_u) - (x_v - x_u) \\ s_u(Y_v - Y_u) - (y_v - x_u) \end{pmatrix}$$

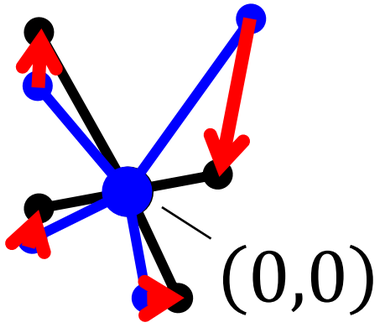


$$[s_u(X_v - X_u) - (x_v - x_u)]^2 + [s_u(Y_v - Y_u) - (y_v - x_u)]^2$$

Measuring Distortion

- compare both (sub-)graphs in ***local reference frame***
- apply (unknown) scale factor $s_u \in \mathbb{R}^+$ to compensate scaling
- ***now*** measure ***squared residuals***

$$\begin{pmatrix} s_u(X_v - X_u) - (x_v - x_u) \\ s_u(Y_v - Y_u) - (y_v - x_u) \end{pmatrix}$$



$$f(x) = \sum_{u \in V} \sum_{v \in \mathrm{Adj}(u)} [s_u(X_v - X_u) - (x_v - x_u)]^2 + [s_u(Y_v - Y_u) - (y_v - x_u)]^2$$

Measuring Distortion

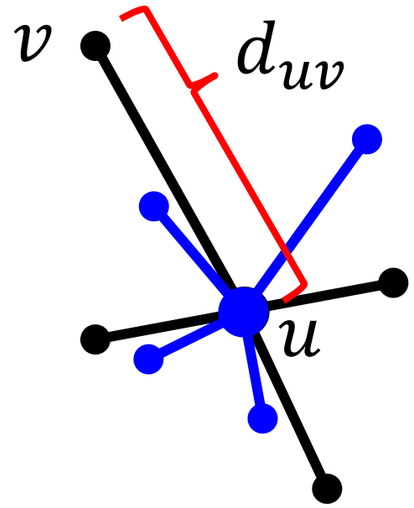
- compare both (sub-)graphs in ***local reference frame***
- apply (unknown) scale factor $s_u \in \mathbb{R}^+$ to compensate scaling
- ***now*** measure ***squared residuals***

$$f(x) = \sum_{u \in V} \sum_{v \in \mathrm{Adj}(u)} \Delta X_{uv}{}^2 + \Delta Y_{uv}{}^2$$

$$\Delta X_{uv} = s_u(X_v - X_u) - (x_v - x_u)$$
$$\Delta Y_{uv} = s_u(Y_v - Y_u) - (y_v - y_u)$$

Measuring Distortion

- compare both (sub-)graphs in ***local reference frame***
- apply (unknown) scale factor $s_u \in \mathbb{R}^+$ to compensate scaling
- ***now*** measure ***squared residuals***



$$f(x) = \sum_{u \in V} \sum_{v \in \mathrm{Adj}(u)} \frac{1}{{d_{uv}}^2} \left({\Delta X_{uv}}^2 + {\Delta Y_{uv}}^2 \right)$$

weighted by squared edge length.

$$\Delta X_{uv} = s_u (X_v - X_u) - (x_v - x_u)$$
$$\Delta Y_{uv} = s_u (Y_v - Y_u) - (y_v - y_u)$$

Measuring Distortion

$$f(x) = \sum_{u \in V} \sum_{v \in \mathrm{Adj}(u)} \frac{1}{d_{uv}^{\,2}} \left(\Delta X_{uv}^{\,2} + \Delta Y_{uv}^{\,2} \right)$$

$$\Delta X_{uv} = s_u (X_v - X_u) - (x_v - x_u)$$
$$\Delta Y_{uv} = s_u (Y_v - Y_u) - (y_v - y_u)$$

variables

constants

Basic Quadratic Program

Minimize

$$f(x) = \sum_{u \in V} \sum_{v \in \mathrm{Adj}(u)} \frac{1}{{d_{uv}}^2} \left({\Delta X_{uv}}^2 + {\Delta Y_{uv}}^2 \right)$$

subject to

$$\Delta X_{uv} = s_u (X_v - X_u) - (x_v - x_u)$$
$$\Delta Y_{uv} = s_u (Y_v - Y_u) - (y_v - y_u)$$

$$X_{\min} \leq x_v \leq X_{\max}$$
$$Y_{\min} \leq y_v \leq Y_{\max}$$

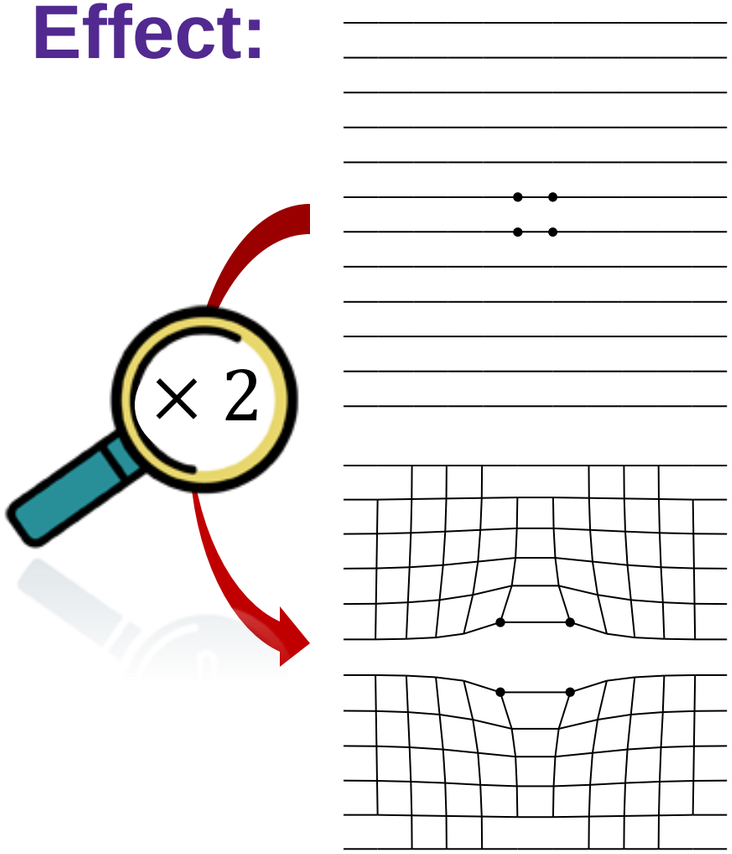
$$s_u = Z$$

for each node u in focus region
($Z =$ ***zoom factor***, e.g., $Z = 3$)

$$\Delta X_{uv} = \Delta Y_{uv} = 0$$

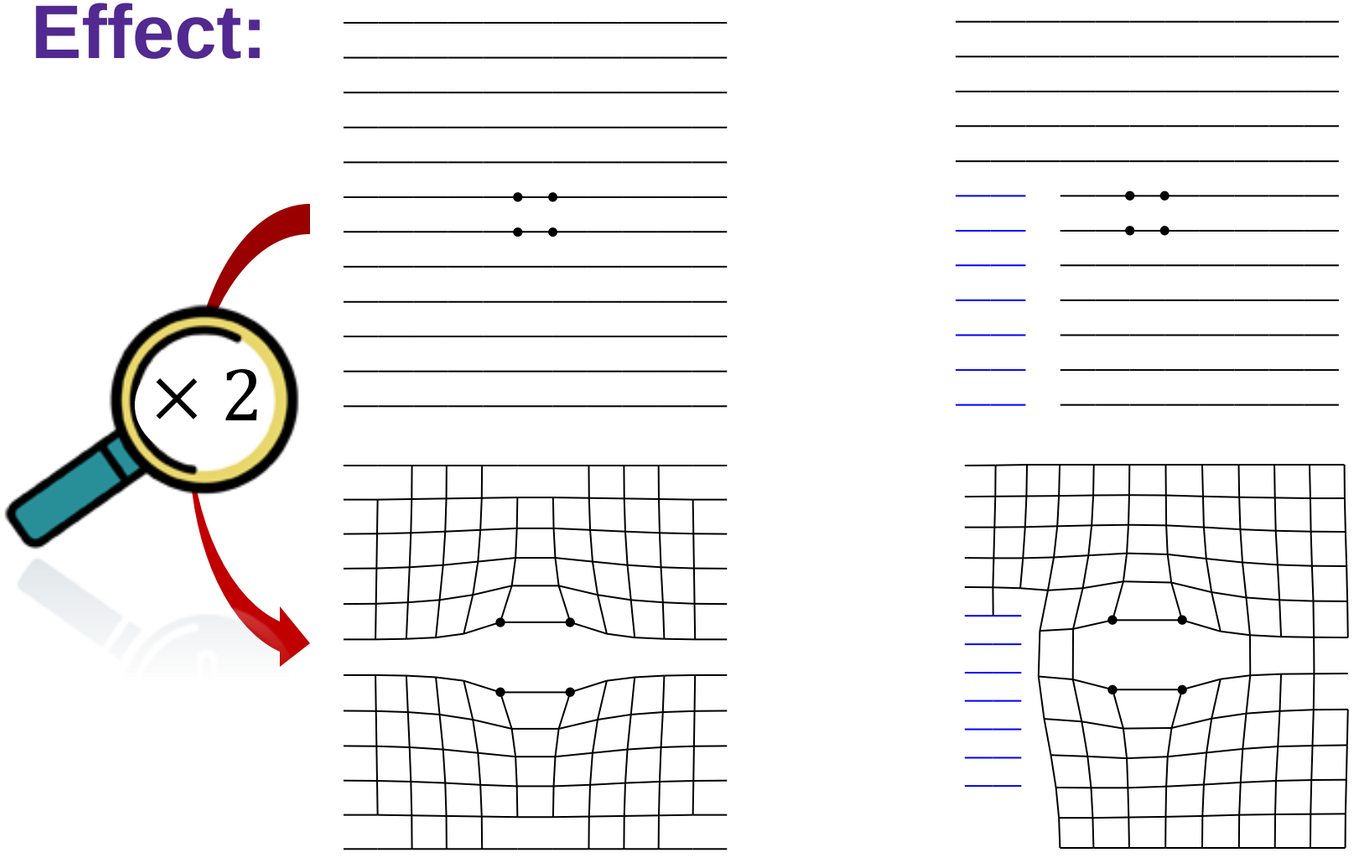
Basic Quadratic Program

Effect:



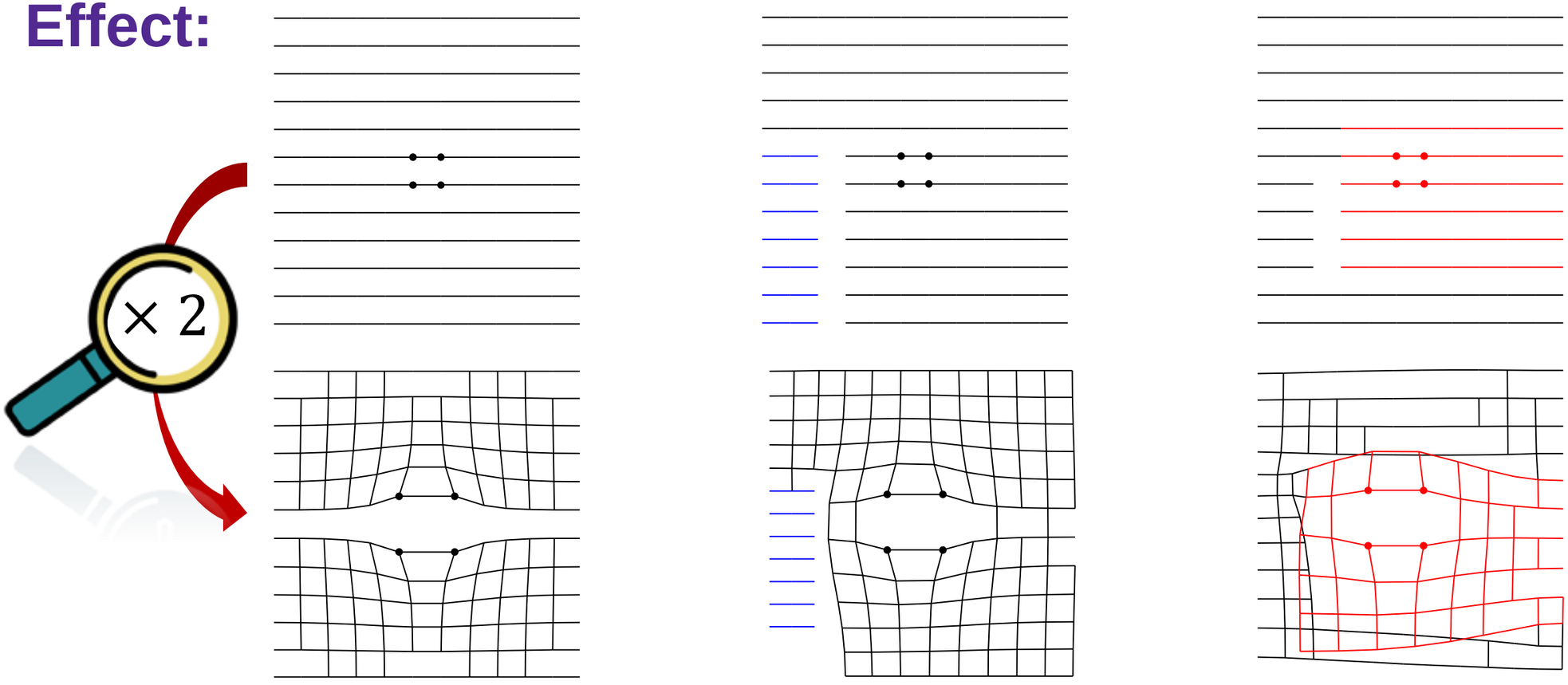
Basic Quadratic Program

Effect:



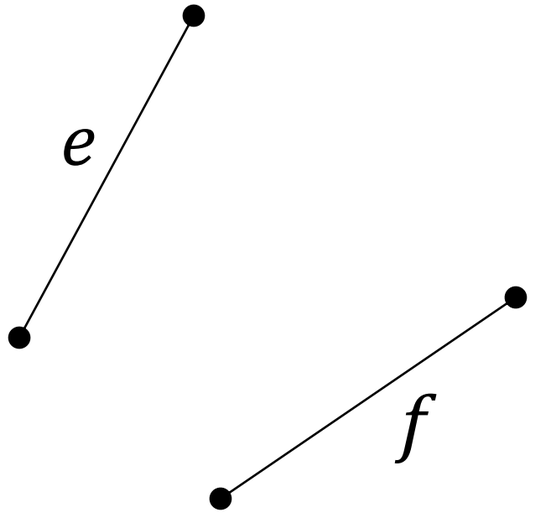
Basic Quadratic Program

Effect:



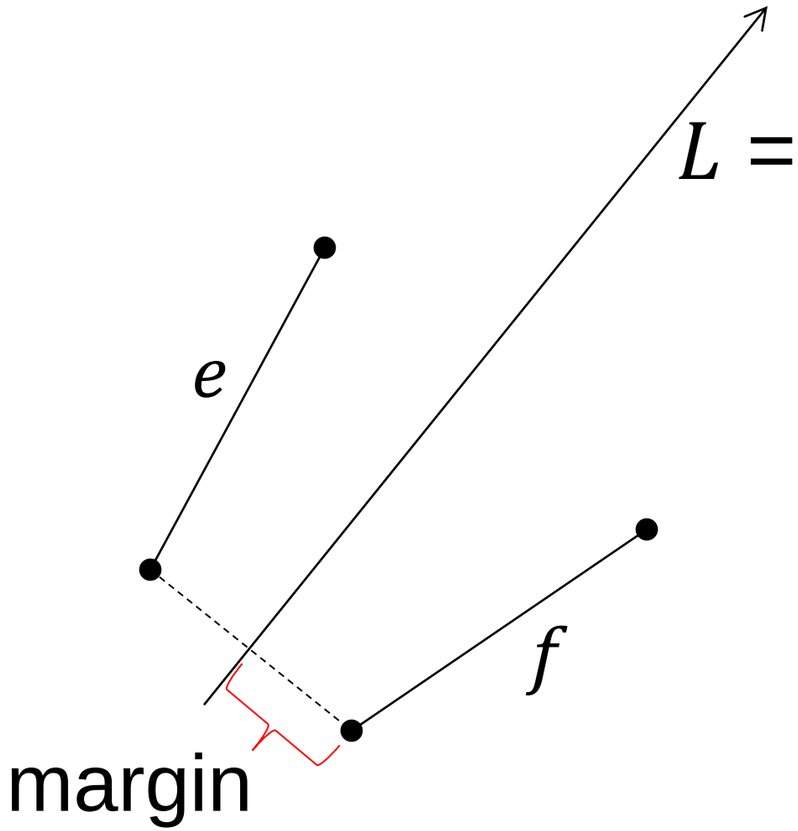
to do: avoid edge crossings

Avoiding Edge Crossings



e and f do not cross

Avoiding Edge Crossings



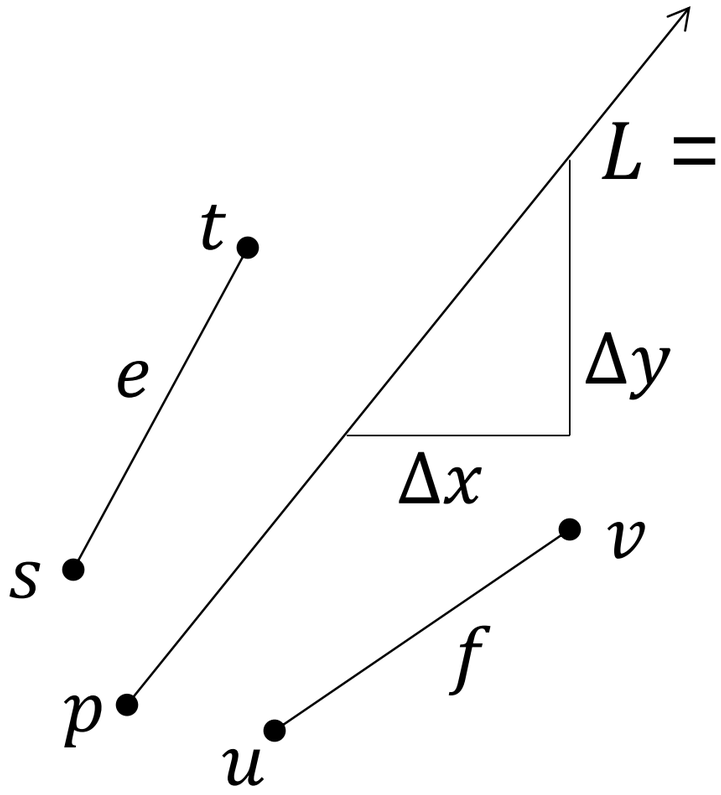
L = maximum-margin line for e and f in input

e and f do not cross

$\Leftarrow$ there exists a line ℓ with

- e to its ***left*** and
- f to its ***right***
- and same slope as L

Avoiding Edge Crossings



L = maximum-margin line for e and f in input

$$(y_s - y_p)/(x_s - x_p) \geq \Delta y/\Delta x$$

$$(y_t - y_p)/(x_t - x_p) \geq \Delta y/\Delta x$$

$$(y_u - y_p)/(x_u - x_p) \leq \Delta y/\Delta x$$

$$(y_v - y_p)/(x_v - x_p) \leq \Delta y/\Delta x$$

Avoiding Edge Crossings

- ***Algorithm:***
  - Solve QP allowing edge crossings
  - Check solution for edge crossings
  - Only add those constraints that forbid crossings found
  - Resolve QP.

$$(y_s - y_p)\Delta x - (x_s - x_p)\Delta y \geq 0$$
$$(y_t - y_p)\Delta x - (x_t - x_p)\Delta y \geq 0$$
$$(y_u - y_p)\Delta x - (x_u - x_p)\Delta y \leq 0$$
$$(y_v - y_p)\Delta x - (x_v - x_p)\Delta y \leq 0$$

These are ***linear*** constraints. 

But their number is $\Theta(|E|^2)$.

Avoiding Edge Crossings

- ***Algorithm:***
  - Solve QP allowing edge crossings
  - Check solution for edge crossings
  - Only add those constraints that forbid crossings found
  - Resolve QP.

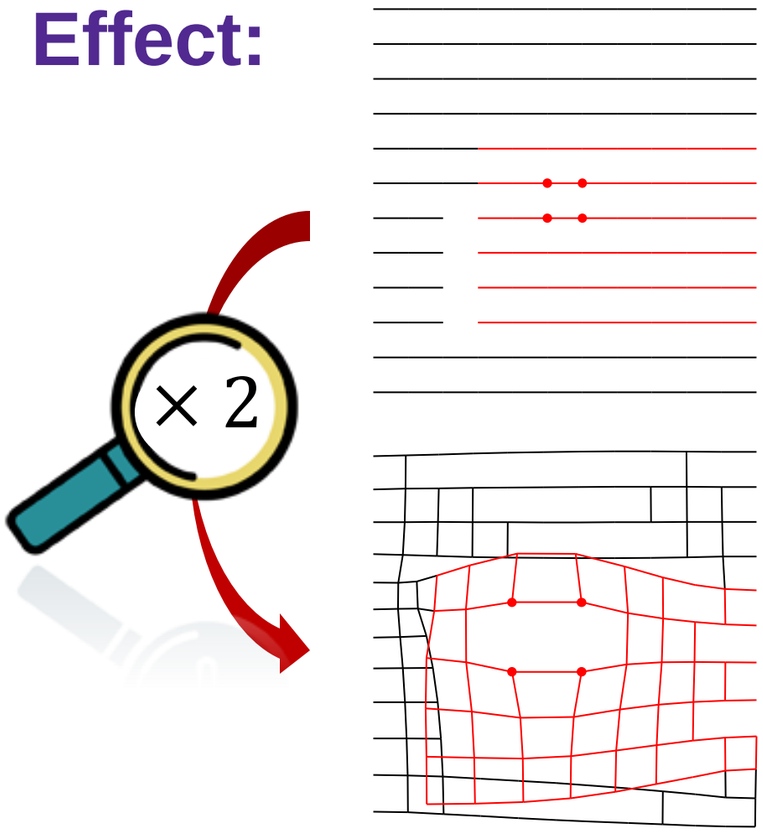
$$(y_s - y_p)\Delta x - (x_s - x_p)\Delta y \geq 0$$
$$(y_t - y_p)\Delta x - (x_t - x_p)\Delta y \geq 0$$
$$(y_u - y_p)\Delta x - (x_u - x_p)\Delta y \leq 0$$
$$(y_v - y_p)\Delta x - (x_v - x_p)\Delta y \leq 0$$

These are ***linear*** constraints. 🙂

But their number is $\Theta(|E|^2)$.

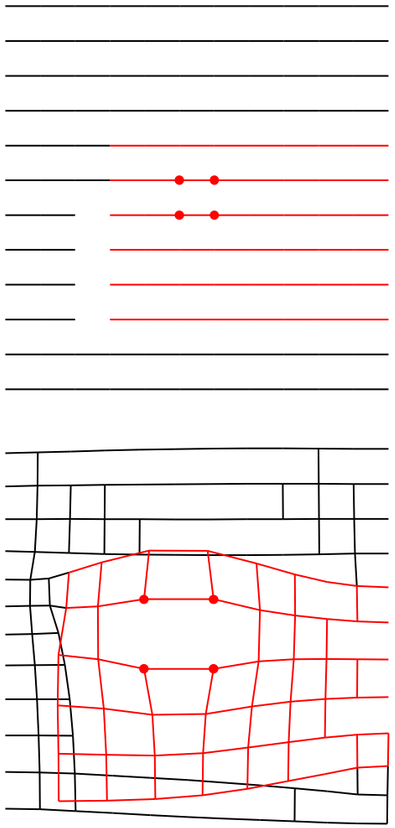
Basic Quadratic Program

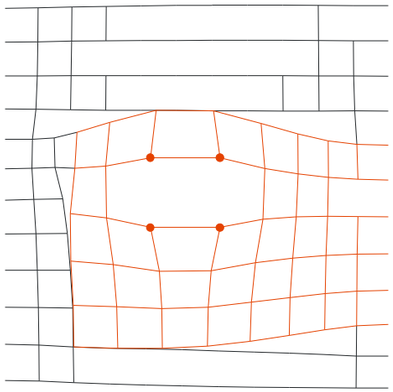
Effect:



Basic Quadratic Program

Effect:



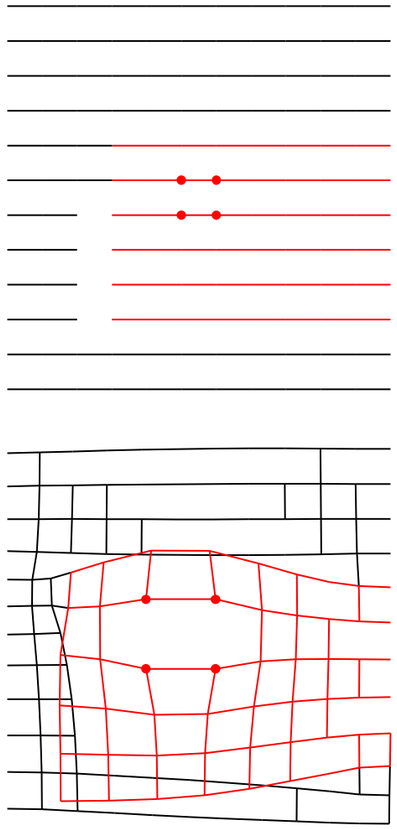


$$(y_s - y_p)\Delta x - (x_s - x_p)\Delta y \geq 0$$

…

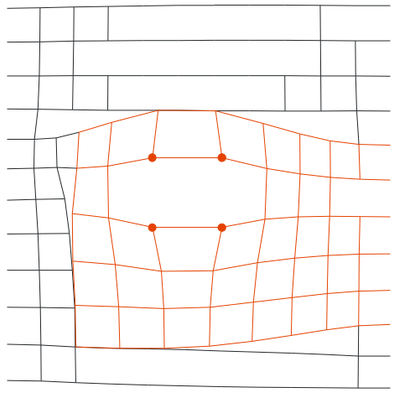
Basic Quadratic Program

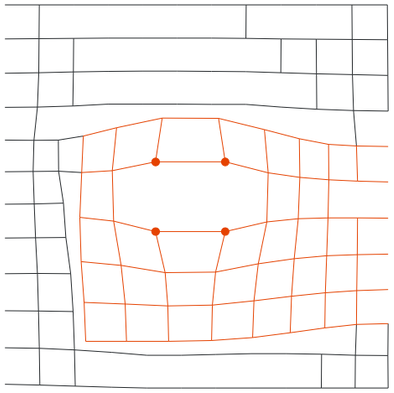
Effect:



$$(y_s - y_p)\Delta x - (x_s - x_p)\Delta y \geq \varepsilon$$

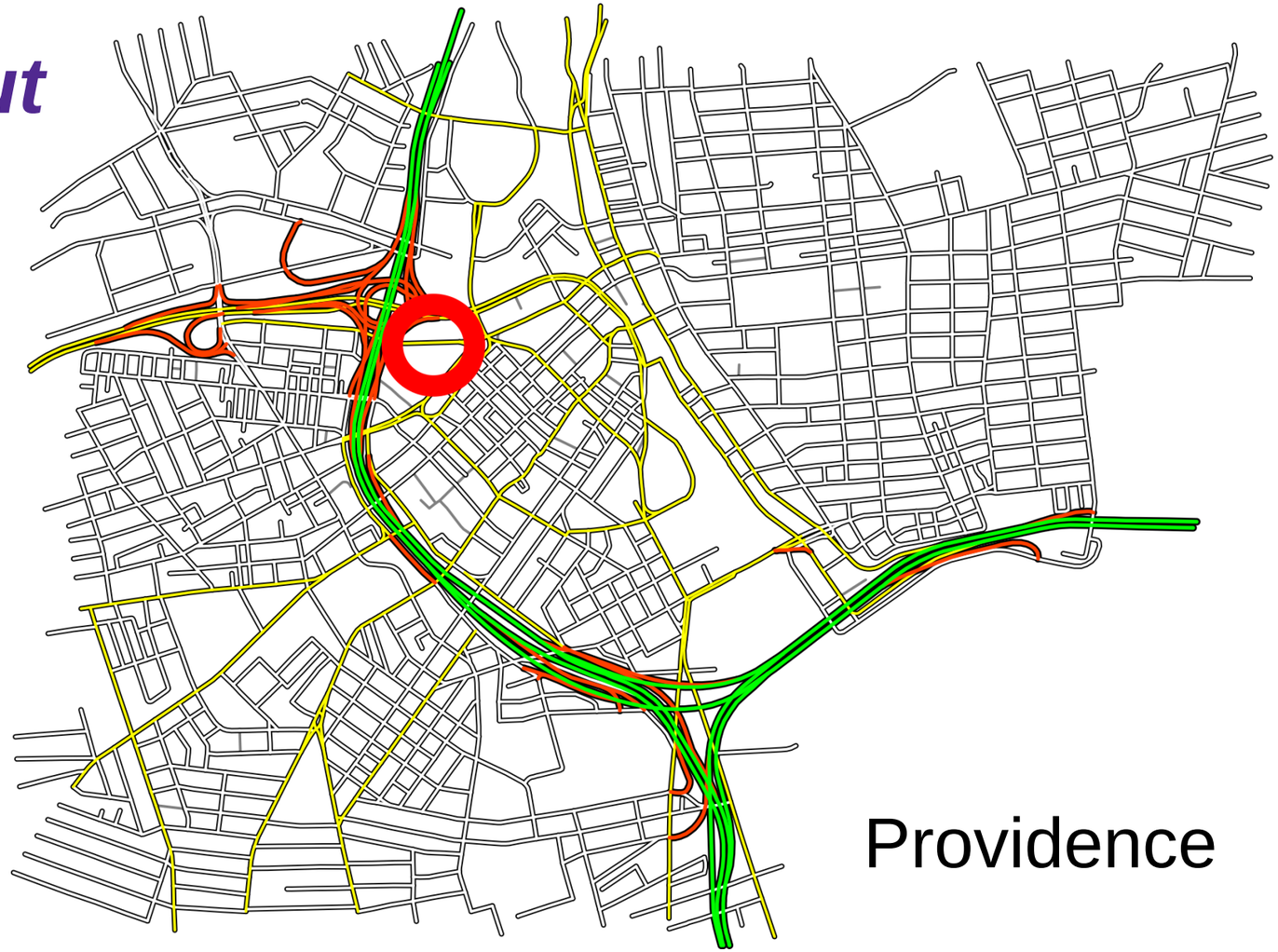
…





Experiments

Input



Providence

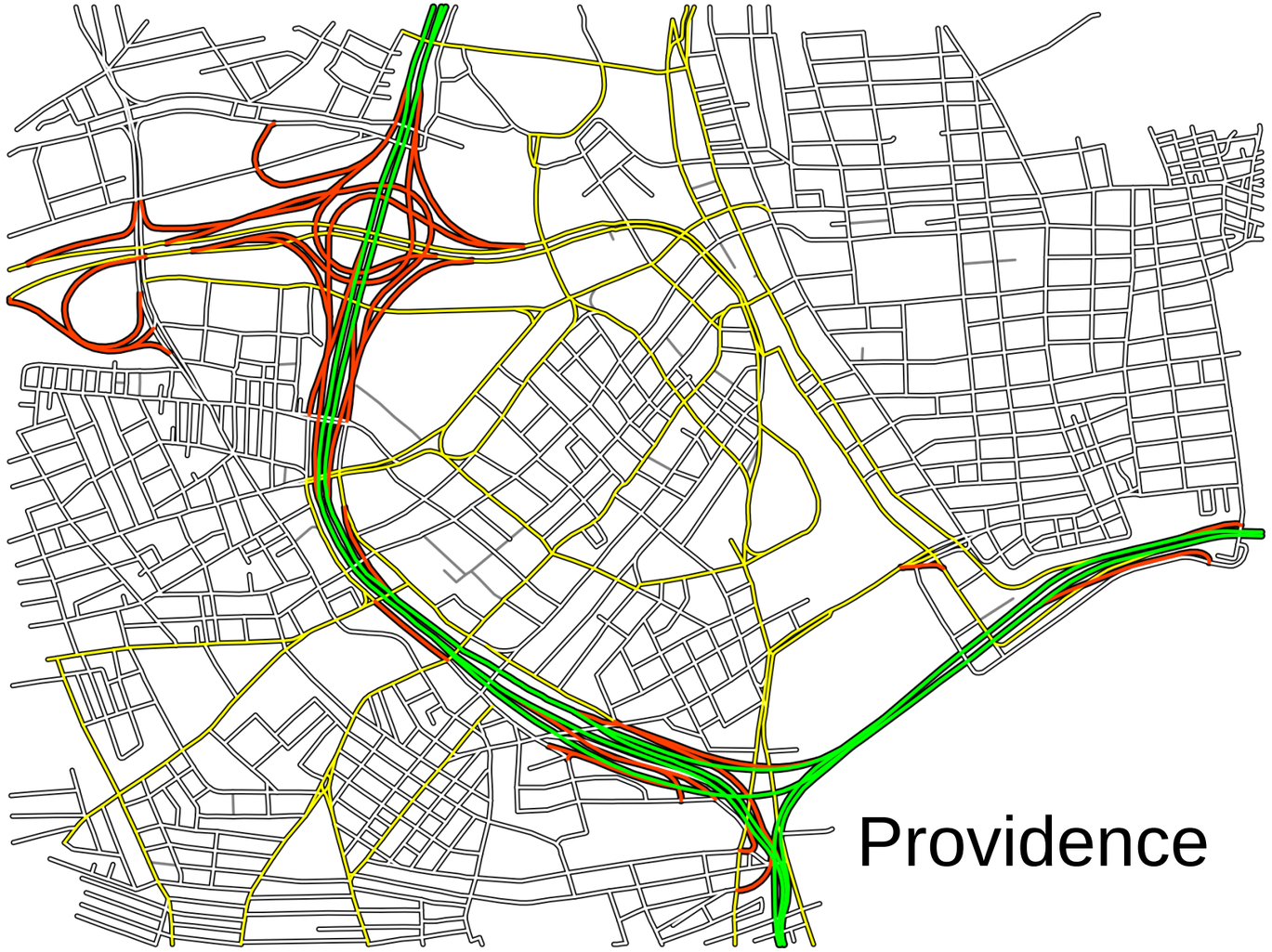
Experiments

Classical Fish-Eye View

Providence

Experiments

Our Result



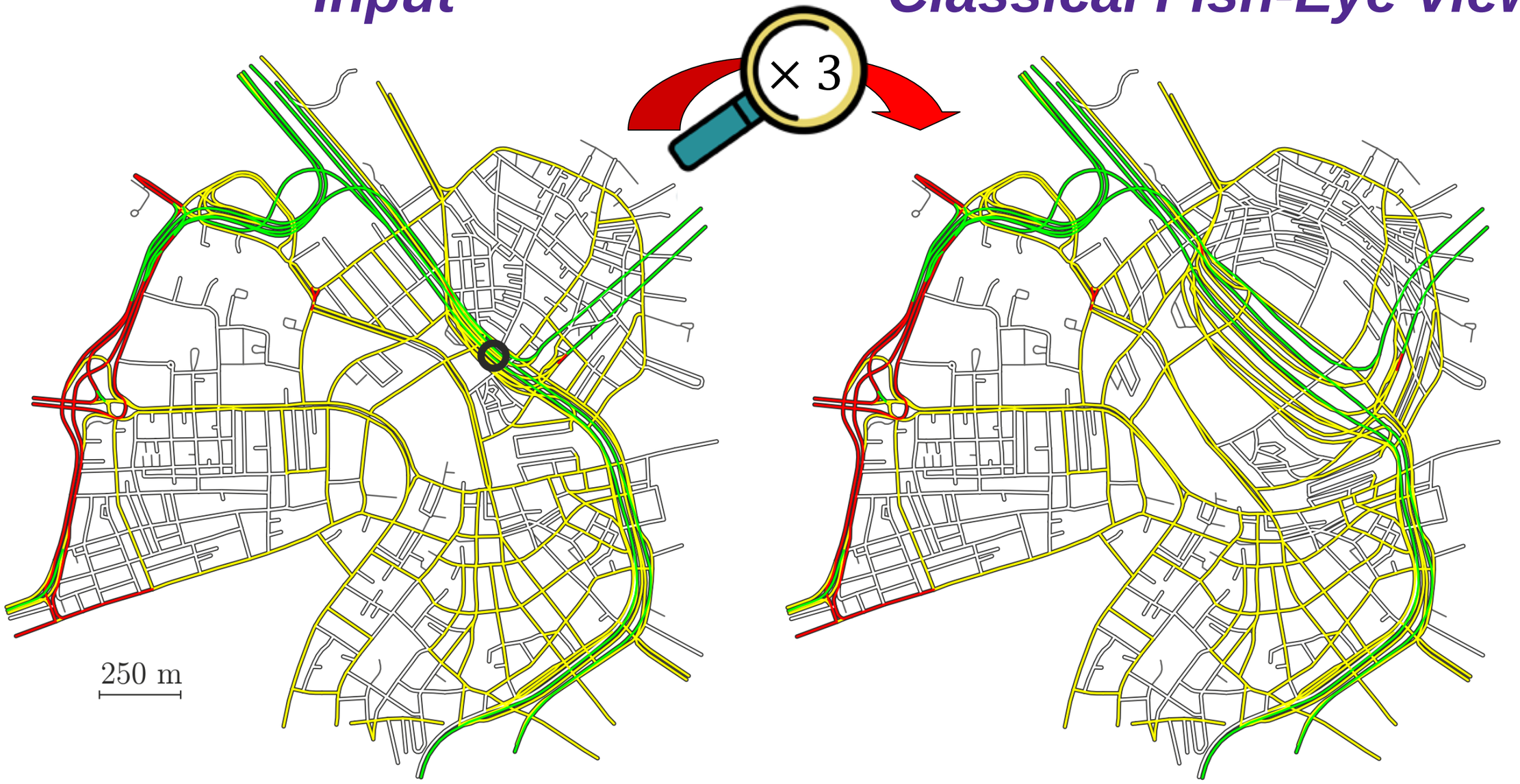
Providence

Experiments

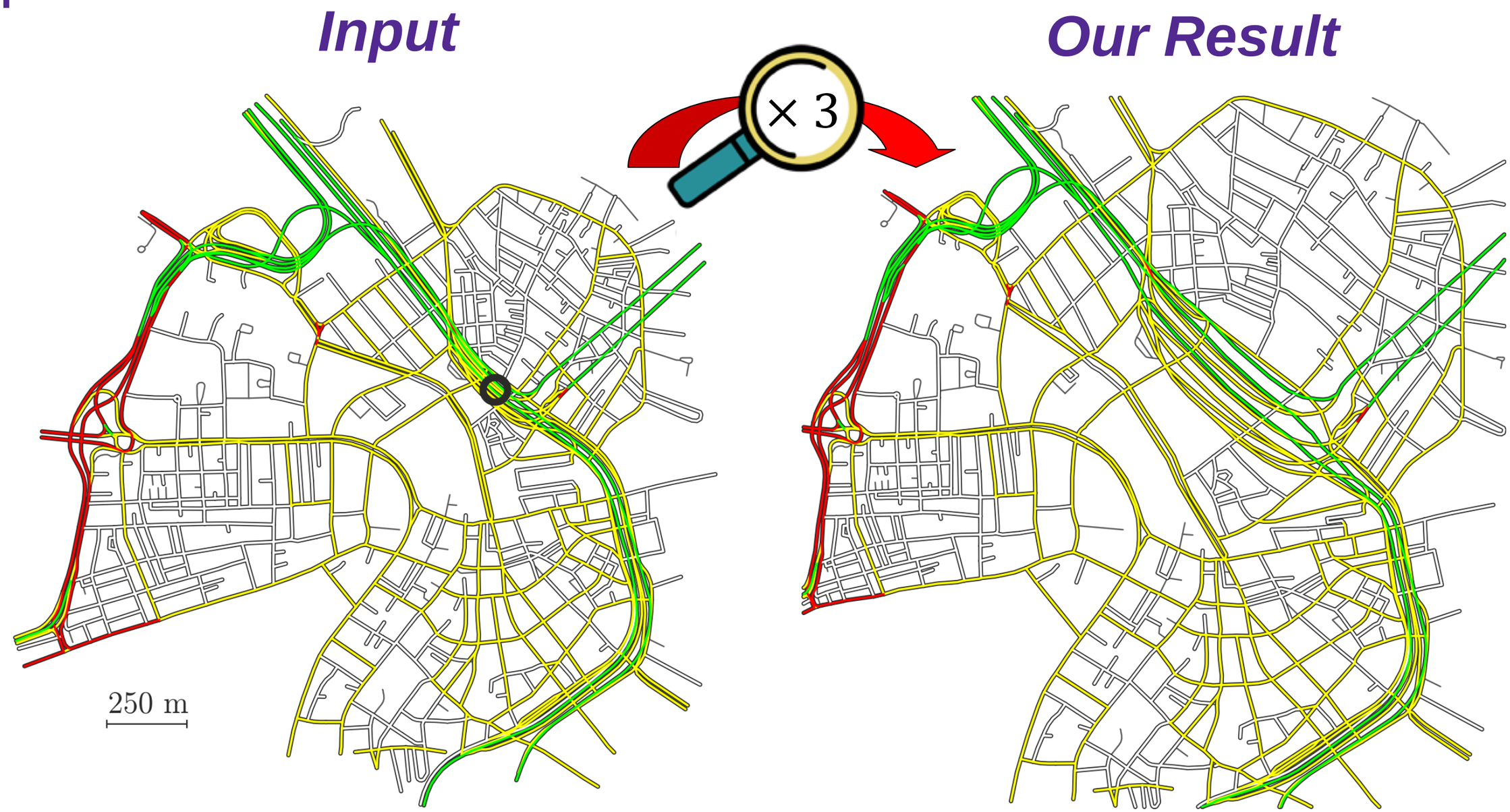
Input

Classical Fish-Eye View

Boston



Experiments



Experiments

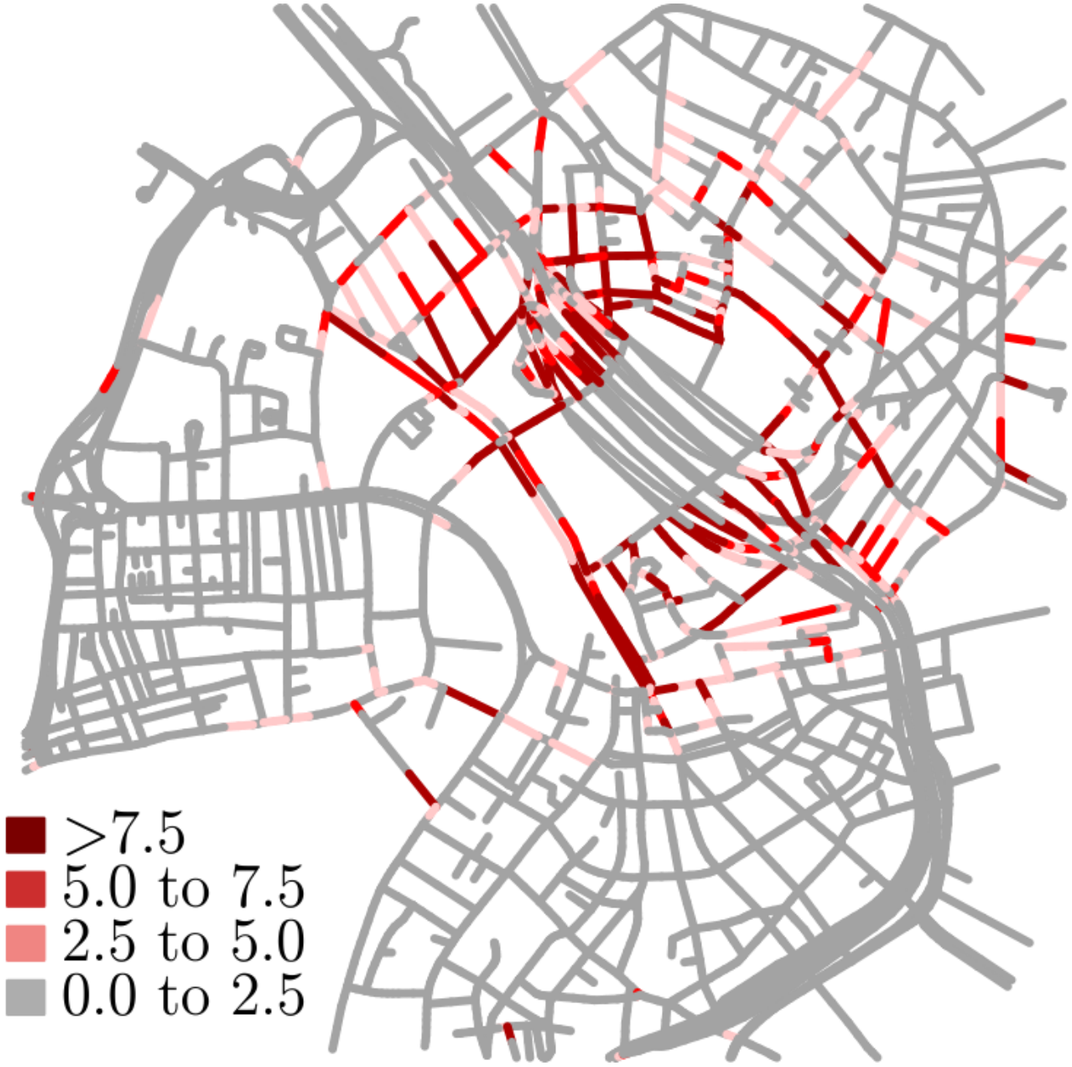
Classical Fish-Eye View

Our Result

Experiments

Classical Fish-Eye View

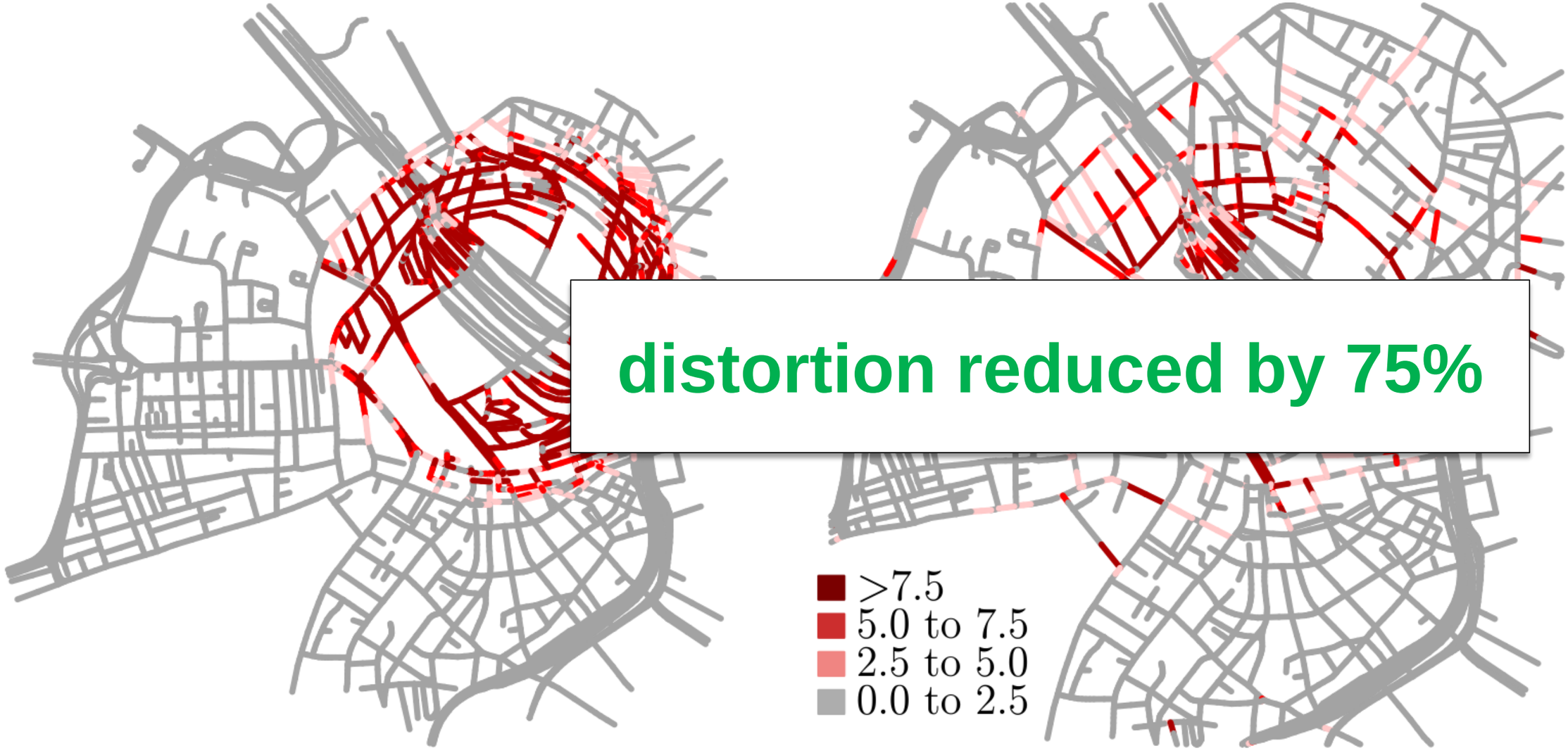
Our Result



Experiments

Classical Fish-Eye View

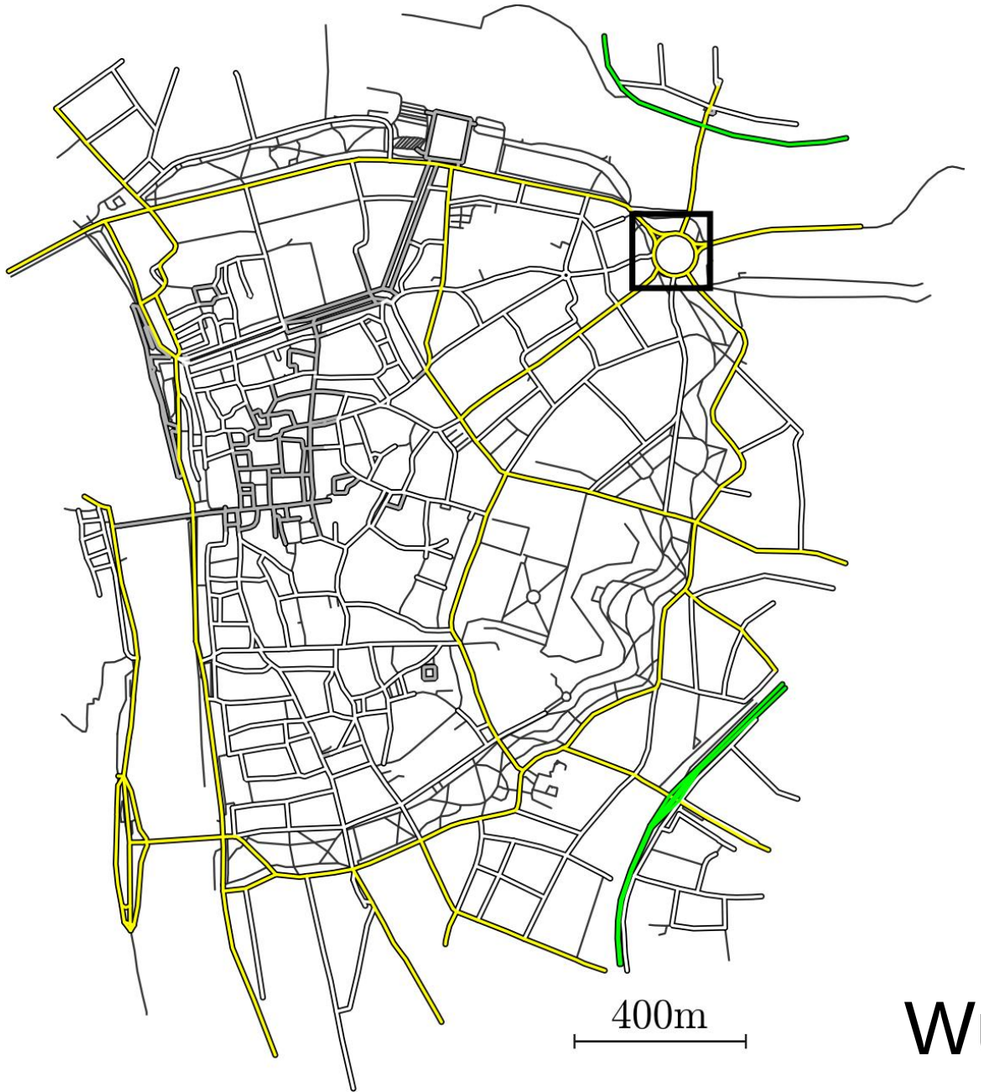
Our Result



distortion reduced by 75%

Experiments

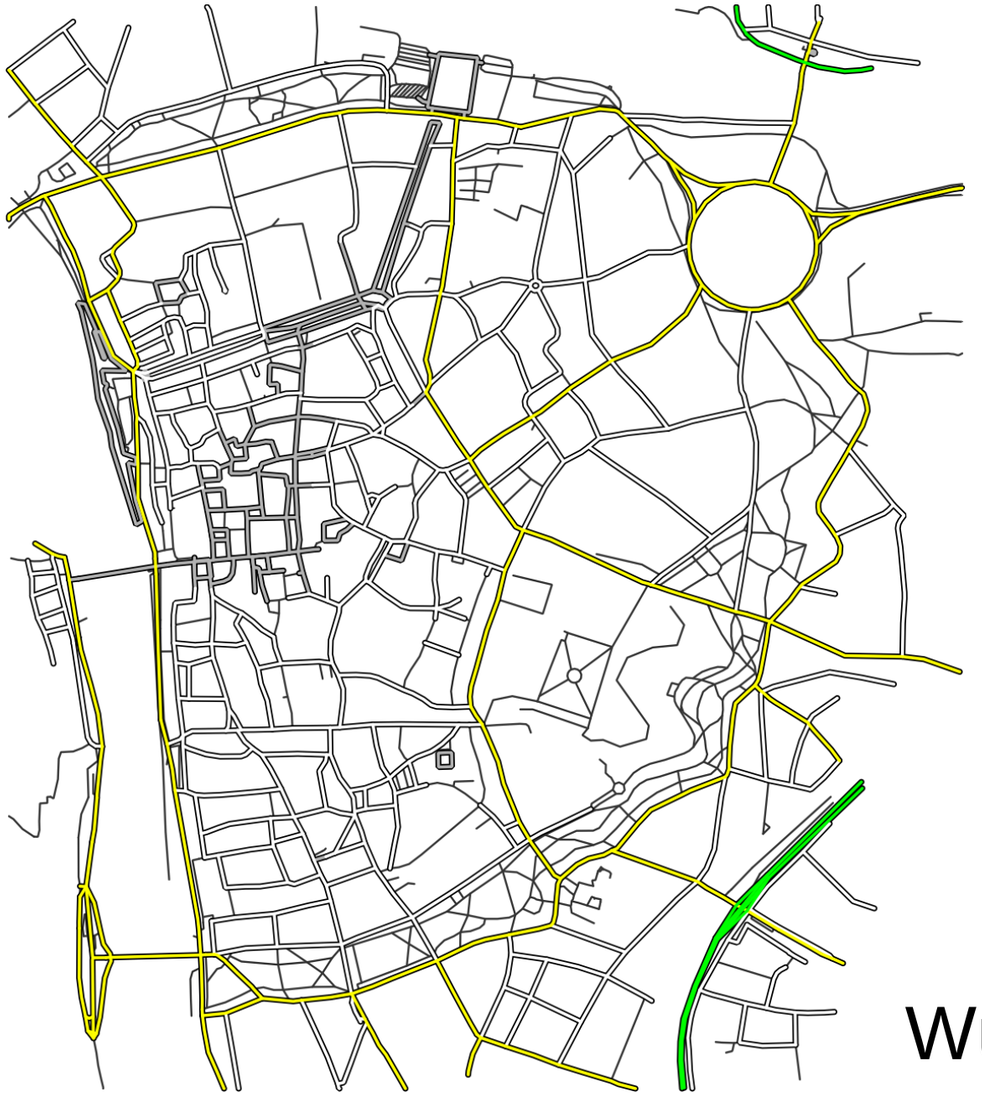
# nodes	# edges	solution time
465	513	1.48 s
2511	***2965***	***7.08 s***
15941	18158	61.70 s



Würzburg

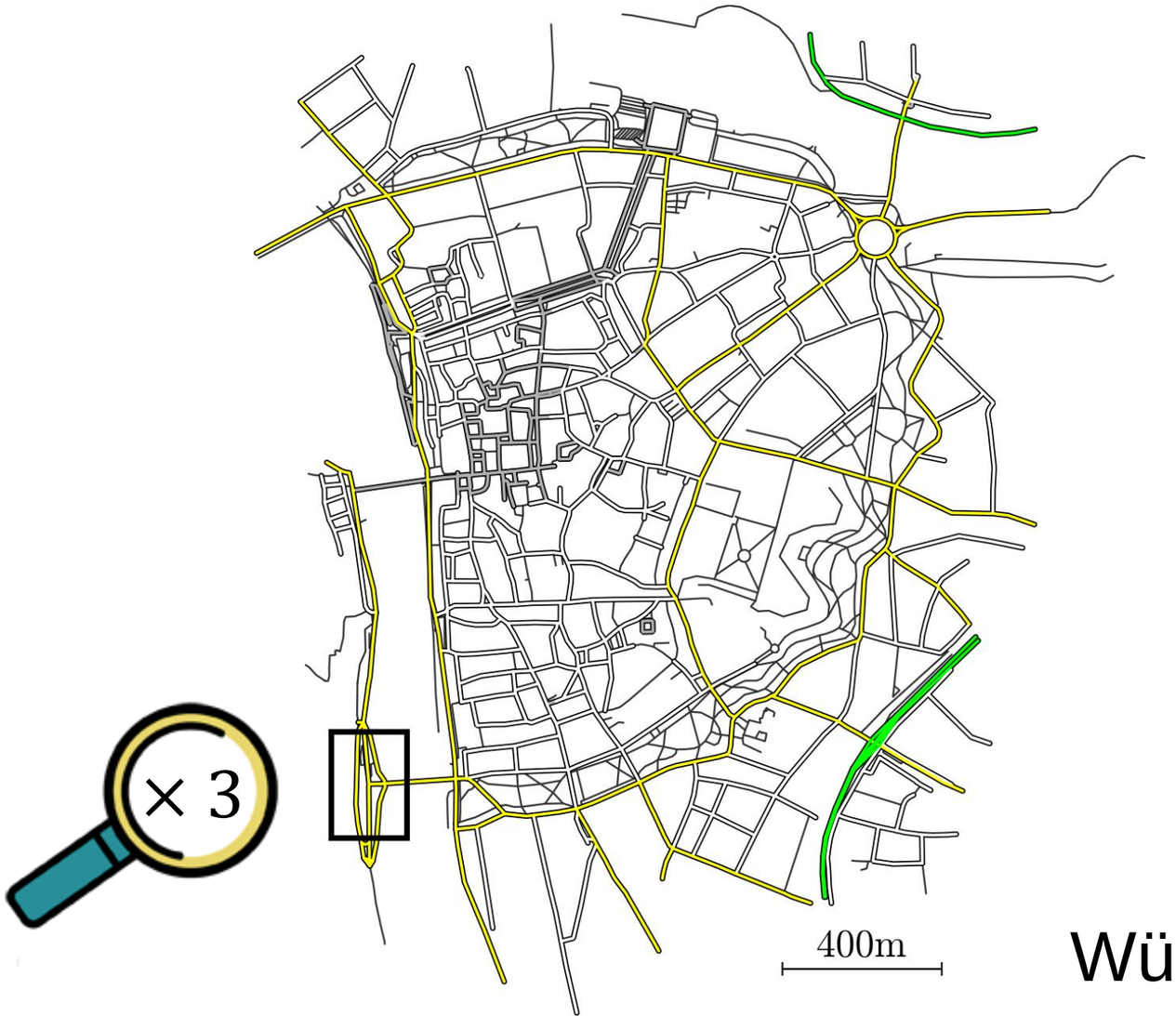
Experiments

# nodes	# edges	solution time
465	513	1.48 s
2511	***2965***	***7.08 s***
15941	18158	61.70 s



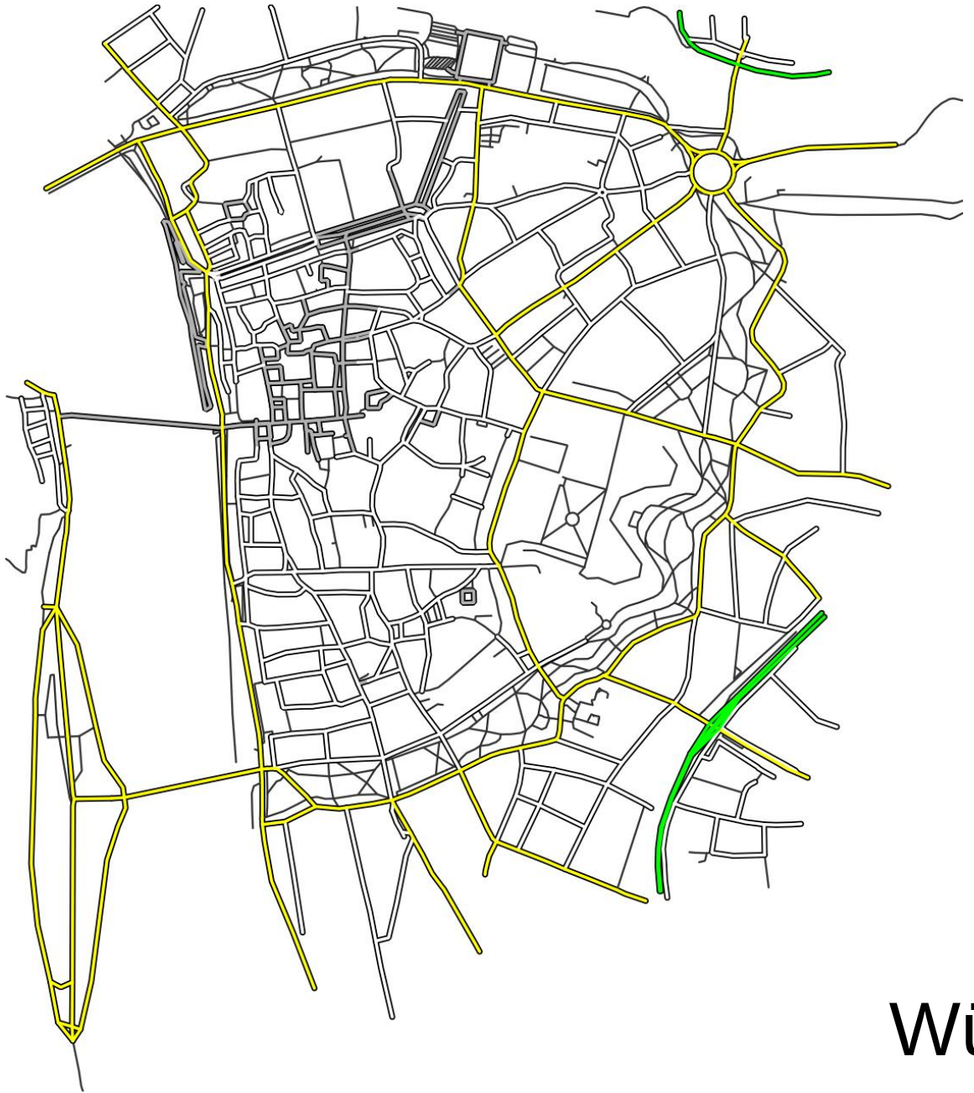
Würzburg

Experiments



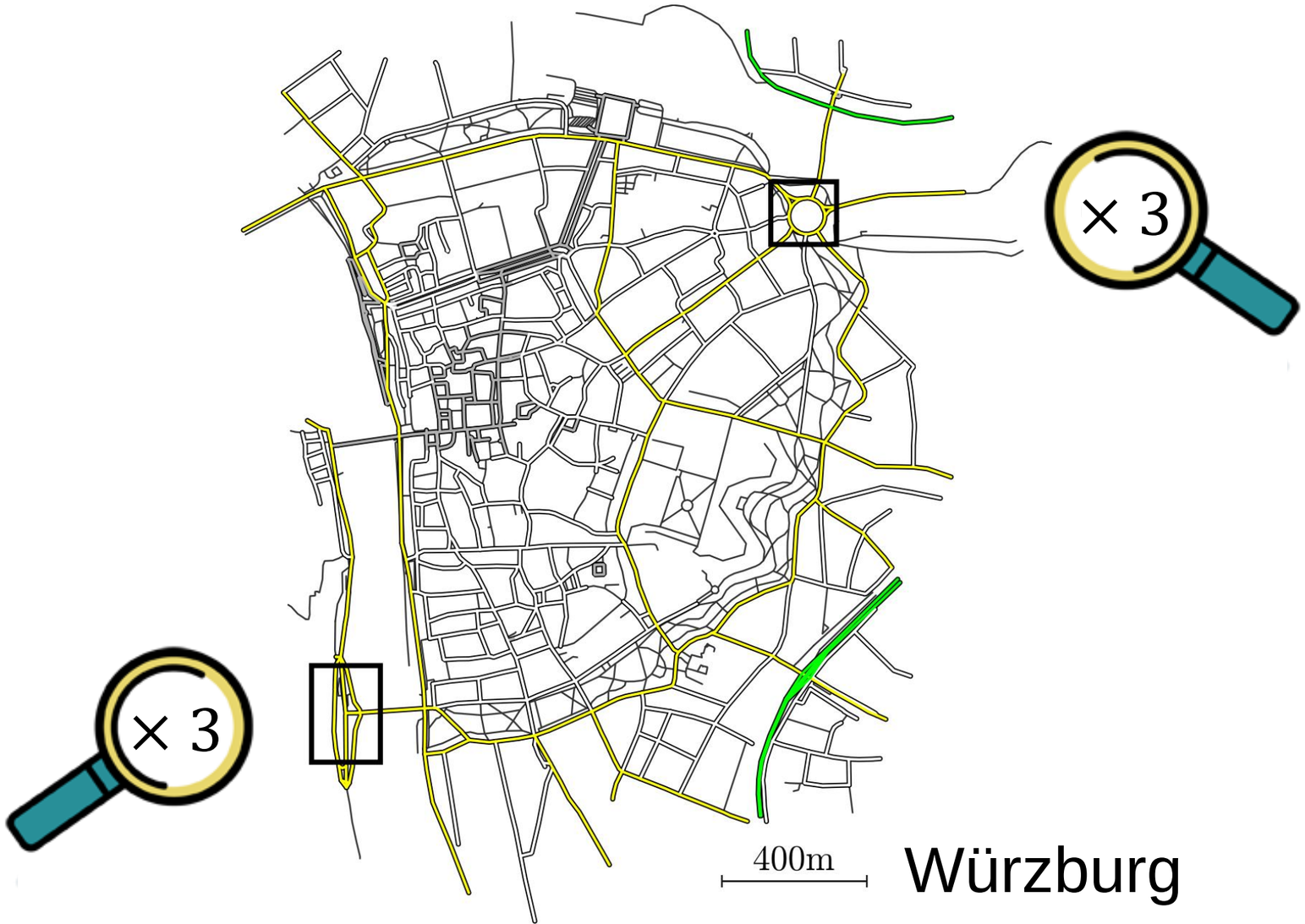
Würzburg

Experiments

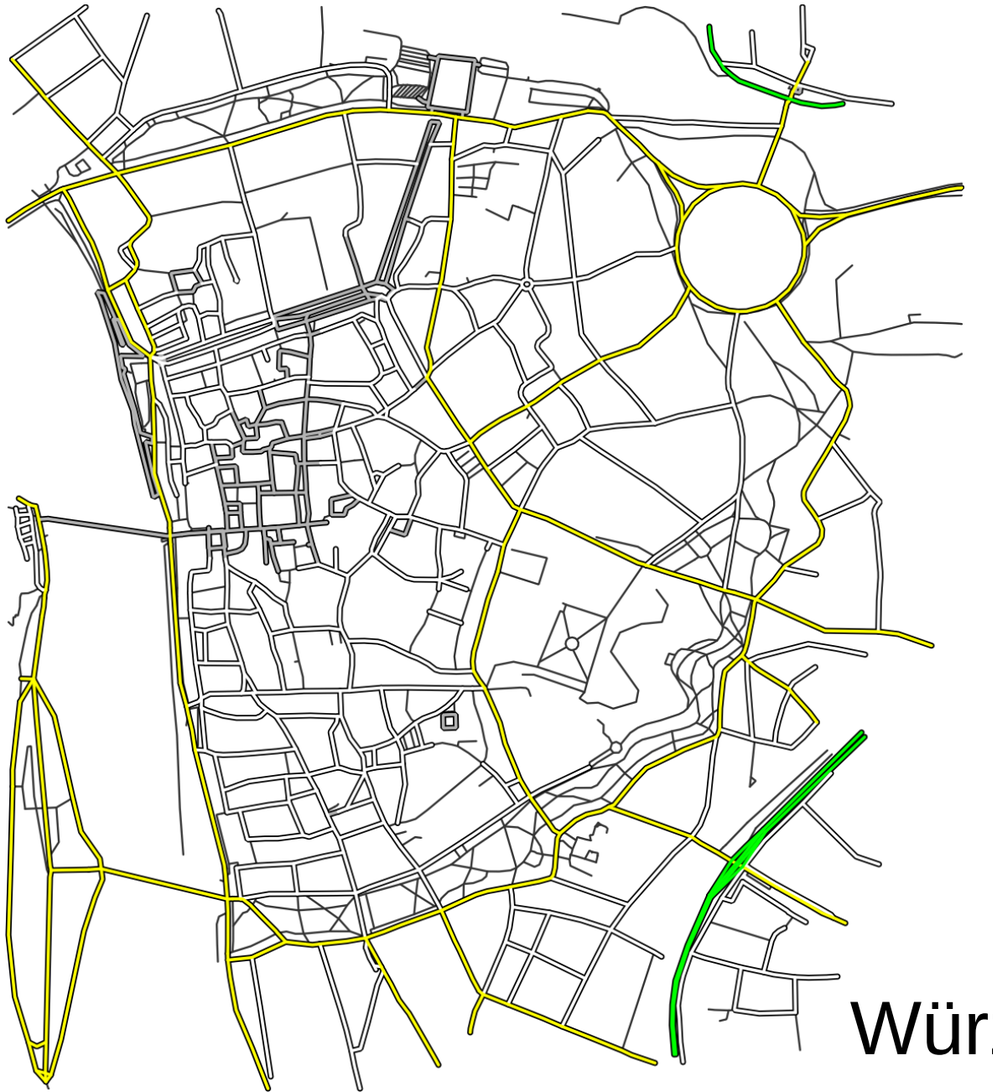


Würzburg

Experiments



Experiments



Würzburg

Conclusion

- New method for ***focus-and-context maps***
  - ***graph*** based
  - ***optimization*** based
  - ***deterministic*** and ***efficient***

  - results ***less distorted*** than with classical fish-eye views
  - ***speed up needed*** for application in real-time systems