Finding Corner Points
Normalized Cross-correlation

Let $w_1 = I_1(x_1 + i, y_1 + j)$ and $w_2 = I_2(x_2 + i, y_2 + j)$, $i = -W, \ldots, W$, $j = -W, \ldots, W$ be two square image windows centered at locations $(x_1, y_1)$ and $(x_2, y_2)$ of images $I_1$ and $I_2$, respectively.

Normalized cross-correlation of $w_1$ and $w_2$ is given by

$$NCC(w_1, w_2) = \frac{(w_1 - w_1) \cdot (w_2 - w_2)}{\|w_1 - w_1\| \|w_2 - w_2\|}$$

where $w_1$ and $w_2$ are treated as vectors. ($a \cdot b$ stands for inner product of vectors $a$ and $b$, $\overline{a}$ for the mean value of vector elements and $\|a\|$ for the 2-norm of vector $a$.)
For two windows whose pixel values differ by a scale factor only NCC will be equal to 1; if the windows are different NCC has value lower than 1.

For two non-zero binary patterns which differ in all pixels NCC is -1.

Normalized cross-correlation corresponds to the cosine of the angle between \( w_1 \) and \( w_2 \); this angle varies between 0° and 180° — the corresponding cosines vary between 1 and -1.

Corner points differ from other points — have low NCC with all other points.
Sum of Squared Differences

Let $w_1 = I_1(x_1 + i, y_1 + j)$ and $w_2 = I_2(x_2 + i, y_2 + j)$, $i = -W, \ldots, W$, $j = -W, \ldots, W$ be two square image windows centered at locations $(x_1, y_1)$ and $(x_2, y_2)$ of images $I_1$ and $I_2$, respectively.

Sum of squared differences of $w_1$ and $w_2$ is given by

$$SSD(w_1, w_2) = \sum_{i=-W}^{W} \sum_{j=-W}^{W} (I_1(x_1 + i, y_1 + j) - I_2(x_2 + i, y_2 + j))^2.$$
For two windows whose pixel values differ by a scale factor only SSD can be very large; if the windows are the same SSD has value 0.

For two non-zero binary patterns which differ in all pixels SSD is \((2W + 1)^2\).

Corner points differ from other points — have high SSD with all other points.
Corner Point Candidates
Corner Point Selection

Cross-correlation results for $9 \times 9$ neighborhoods of the three image points on the previous slide.

Corresponding similarity functions shown from an $80^\circ$ viewing angle. Two outside points are good feature point candidates, while the center point is not.
Selected Corner Points

Selected feature points for 1/2 resolution: each point is shown with its $9 \times 9$ neighborhood.
Selected Corner Points

Selected feature points for 1/4, and 1/8 resolutions: each point is shown with its $9 \times 9$ neighborhood.
Problems with NCC

We need something more efficient for a practical corner detection algorithm.

An NCC-based algorithm detects points that are different from their neighborhood. We need a measure of “cornerness”.

Alternative: use a measure based on local structure.
Consider the spatial image gradient $[E_x, E_y]^T$, computed for all points $(x, y)$ of an image area (neighborhood). The matrix $M$, defined as

$$M = \begin{pmatrix} \sum E_x^2 & \sum E_x E_y \\ \sum E_x E_y & \sum E_y^2 \end{pmatrix}$$

where the sums are taken over the image neighborhood, captures the geometric structure of the gray level pattern. Note that the sums can be replaced by Gaussian/binomial smoothing. Binomial smoothing will improve the localization of corner points. WHY?
$M$ is a symmetric matrix and can therefore be diagonalized by rotation of the coordinate axes, so with no loss of generality, we can think of $M$ as a diagonal matrix:

$$M = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix},$$

where $\lambda_1$ and $\lambda_2$ are the eigenvalues of $M$.

We can choose $\lambda_1$ as the larger eigenvalue so that $\lambda_1 \geq \lambda_2 \geq 0$.

If the neighborhood contains a corner, then we expect $\lambda_1 > \lambda_2 \geq 0$, and the larger the eigenvalues, the stronger (higher contrast) their corresponding edges.

A corner is identified as two strong edges; therefore as $\lambda_1 > \lambda_2$, a corner is a location where $\lambda_2$ is sufficiently large.
Solve: $det(M - \lambda I) = 0$ to obtain $\lambda_1$ and $\lambda_2$.

There are three cases:

**No structure:** (smooth variation) $\lambda_1 \approx \lambda_2 \approx 0$

**1D structure:** (edge) $\lambda_2 \approx 0$ (direction of edge), $\lambda_1$ large (normal to edge)

**2D structure:** (corner) $\lambda_1$ and $\lambda_2$ both large and distinct

The eigenvectors $\vec{n}$ of $M$ ($M\vec{n} = \lambda\vec{n}$) correspond to the direction of smallest and largest change at $(x, y)$. 
Computing Cornerness

**METHOD 1:**

Identify the corner points using the measure described earlier.

Consider all image points for which $\lambda_1 \geq \tau$ and $\lambda_1/\lambda_2 \leq \kappa$.
(For example, use $\tau$ equal to one-twentieth of the maximum $\lambda_1$ in the entire image and we use $\kappa = 2.5$.)

Decreasing $\tau$ or increasing $\kappa$ will typically result in a larger number of candidate points.

**METHOD 2:**

Consider all image points for which $\lambda_1 \geq \tau$. 
Example: selected corner points
Selected corner points for full resolution.
Selected corner points for 1/2 and 1/4 resolutions.