A - trimmed : First, admissible \( x \% \) of Carpest

\( d(k, r) = 1 \quad A \quad k = 0 \) normal (unweighted)

Where \( d(\cdot, \cdot) \) is a distance

Weighted \( d(\cdot, \cdot) \) times

Weighted : Pixel values in kernel are

Pixel value \( = \) median value inside kernel

\( \frac{\text{Weighted Median} / \text{Median}}{\text{Same for Mean Filters}} \)

Still - Filters away high frequency noise

Non - Linear smoothing preserves edges while ir can
To find pixels that are "too much" from the current pixel value:

\[
\begin{align*}
    r_{\text{exp}}(k, z) &= \exp\left(-\|f(k, z) - f(0, 0)\|^2\right) \\
    r_{\text{Gaussian}}(k, z) &= \exp\left(-\frac{(i-k)^2 + (j-z)^2}{2\sigma^2}\right)
\end{align*}
\]

Instead, it can be seen:

same as $\alpha$-transmission, but $\alpha$ is not fixed
<table>
<thead>
<tr>
<th>Range Filter</th>
<th>Domain Filter</th>
<th>q-mean = 4.6</th>
<th>Median = 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2 0.4 1.0 0.8 0.4</td>
<td>0.1 0.3 0.4 0.1</td>
<td>6 8 7 4</td>
<td>6 8 7 4</td>
</tr>
<tr>
<td>0.2 0.2 0.8 0.8 1.0</td>
<td>0.3 0.6 0.8 0.6 0.3</td>
<td>7 6 4 8 3</td>
<td>7 6 4 3</td>
</tr>
<tr>
<td>0.3 0.0 0.0 0.0 0.0</td>
<td>0.4 0.0 0.0 0.0 0.0</td>
<td>1 2 1 2 4</td>
<td>1 2 1 2</td>
</tr>
<tr>
<td>0.3 0.4 0.0 0.0 0.0</td>
<td>0.1 0.3 0.4 0.1 0.2</td>
<td>2 1 0 1 2</td>
<td>2 1 0 1</td>
</tr>
</tbody>
</table>
to remove the salt noise because the noisy pixels are too different from their neighbors.

C) Gaussian blurring: (g) Median blurred; (h) Bilaterally blurred. Note that the bilateral blur

Salt and pepper: (e) Median blurred; (d) Bilaterally blurred; (c) Original image with salt and pepper noise; (b) Gaussian blur; (a) Median blur.
\[ d((x, y), (k, e)) = \min\ a(x - k, y - e) \]

Euclidean distance

\[ d((0, 0), (k, e)) = \sqrt{k^2 + e^2} \]

Manhattan/City Block distance

- Requires two-pass passes algorithm
- Computes two distances to curves/sets of points
Connected vs. Disconnected

Connected Components

4 neighbors vs. 8 neighbors

Connected Components (Abelian)
Geometric Transformations

DOMAIN OF THE IMAGE

Transformation

RANGE OF THE IMAGE

Transformation

\[ g(x) = f (x, y) \]

\[ f (x) = g (x, y) \]

Filters & Other Transformations
<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th># DoF</th>
<th>Preserves</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>projective</td>
<td>(\hat{H})</td>
<td>3×3</td>
<td>8</td>
<td><img src="image" alt="Parallelogram" /></td>
</tr>
<tr>
<td>affine</td>
<td>(A)</td>
<td>2×3</td>
<td>6</td>
<td><img src="image" alt="Parallelogram" /></td>
</tr>
<tr>
<td>similarity</td>
<td>([sR, t])</td>
<td>2×3</td>
<td>4</td>
<td><img src="image" alt="Square" /></td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>([R, t])</td>
<td>2×3</td>
<td>3</td>
<td><img src="image" alt="Square" /></td>
</tr>
<tr>
<td>translation</td>
<td>([I, t])</td>
<td>2×3</td>
<td>2</td>
<td><img src="image" alt="Square" /></td>
</tr>
</tbody>
</table>

- **Ratios**: Between distances
- **Preserves**: Ratios, Parallels, Angles, Lengths, Orientation
Figure 3.47

(a) Inverse warping algorithm: (a) a pixel in image $y$ is sampled from its corresponding location $x$.

(b) Forward warping algorithm: (a) a pixel in image $y = x$ is copied to its corresponding location $x$. 

Figure 3.46
the above procedure described by performing a single data normalization, such as in the paper proposed in [12], prior to computing the contribution of each pixel, the contribution of each cluster, and the cluster results can be mapped across clusters, such as in pixels or in words across documents, and some feedback which may be associated with the model evaluation.

In [17], the right singular vector of \( L \) is associated with the smallest singular value (or eigenvalue), the eigenvector of \( L \) associated with the largest eigenvalue of \( L \), is

\[
\begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]

Where we are given \( 1 \) in points, we have unknown solutions, which can be written in matrix equation as

\[
X = \left[ \begin{array}{c}
0 \\
0 \\
0
\end{array} \right]
\]