FOURIER FEATURES LET NETWORKS LEARN HIGH FREQUENCY FUNCTIONS IN LOW DIMENSIONAL DOMAIN

INTRODUCTION

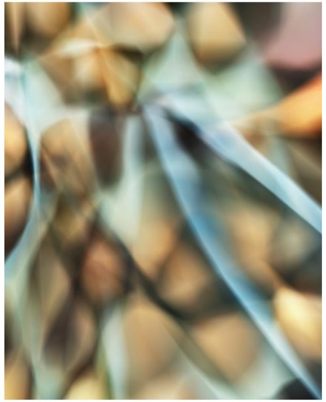
- High Frequency Functions - Images.
- Low Dimensional Domain - In R^3 dimensional space
- MLPs have difficulty learning high frequency functions, a phenomenon referred as “spectral bias”
- The paper states that the researchers came up with a fourier feature mapping that enables neural networks to overcome the bias..
- It is proved by using tools from the Neural Tangent Kernel Literature.
- Fourier Feature Mapping transforms the kernel into a **stationary and tunable** one.

SPECTRAL BIAS

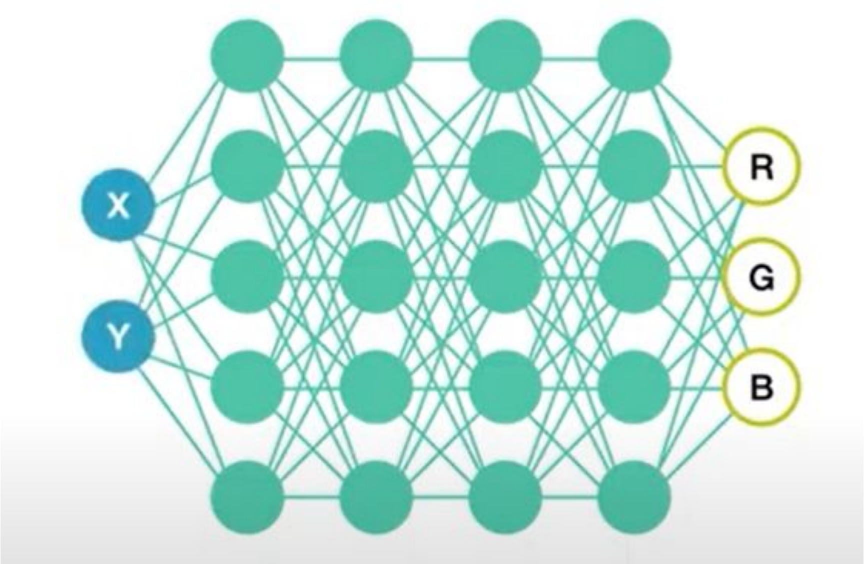
https://arxiv.org/abs/1806.08734 - On the Spectral Bias of Neural Networks, MILA

- While neural networks can approximate arbitrary functions, the paper provides evidence that low frequencies are learnt first during training.
- Implies, the learning bias of deep neural networks manifests itself as a frequency dependent learning speed.

Ground Truth



Output w/o Mapping

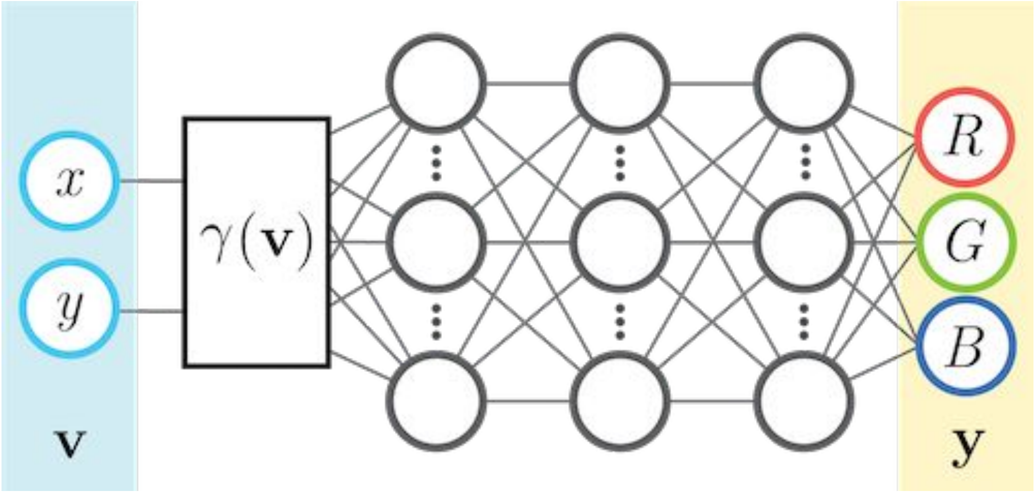


MLP representing an image. Input is a single 2D input pixel coordinate and output is a single RGB colour

MLP Output Ground Truth

(a) Coordinate-based MLP (b) Image regression $(x,y) \to$ RGB (c) 3D shape regression $(x,y,z) \to$ occupancy (d) MRI reconstruction $(x,y,z) \to$ density (e) Inverse rendering $(x,y,z) \to$ RGB, density

- Coordinate-based MLPs, take low-dimensional coordinates as inputs (typically points in R^2 for 2d images, R^3 for 3d) and are trained to output a representation of shape, density, and/or color at each input location



Passing Input Pixel Coordinates through a fourier mapping denoted by γ(v)

COORDINATE MAPPING

1. No mapping: $\gamma(v)=v$
2. Basic mapping: $\gamma(v)=[\cos(2\pi v),\sin(2\pi v)].T$
3. Gaussian Fourier feature mapping: $\gamma(v)=[\cos(2\pi Bv),\sin(2\pi Bv)].T$

B is sampled from $N(0, \sigma^2)$

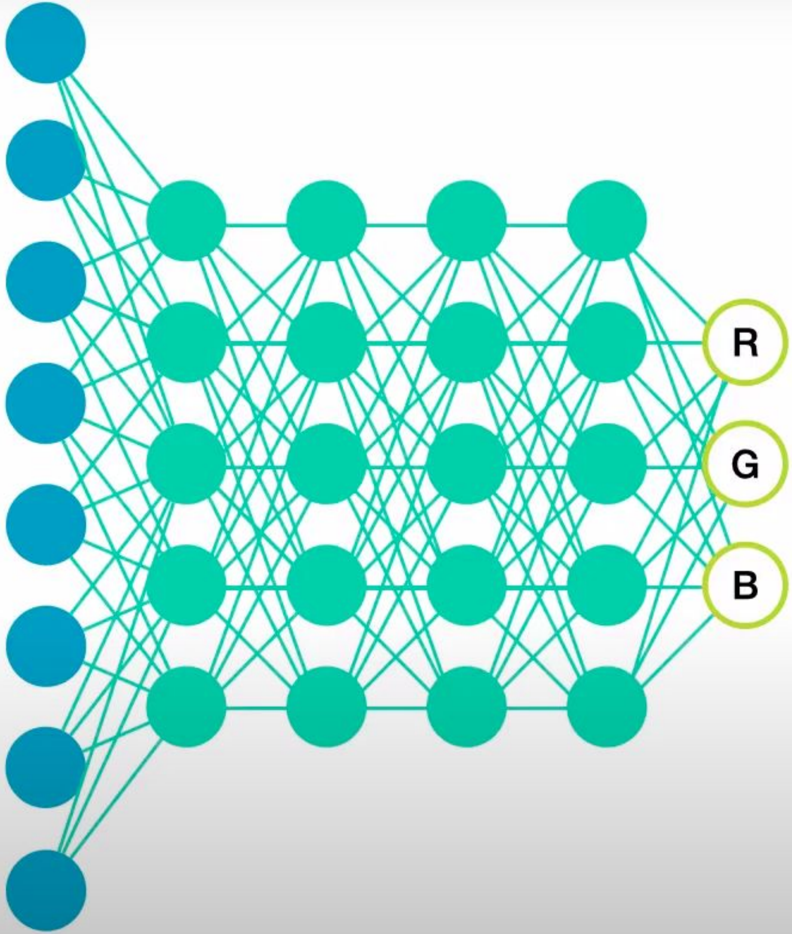
It was also demonstrated that B can be sampled from other distributions such as Uniform Log and Laplacian and the results would still be the same.

Fourier feature mapping

$$\sin\left(\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \\ b_{3,1} & b_{3,2} \\ b_{4,1} & b_{4,2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}\right)$$

$$\cos\left(\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \\ b_{3,1} & b_{3,2} \\ b_{4,1} & b_{4,2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}\right)$$

Entries of B matrix sampled randomly

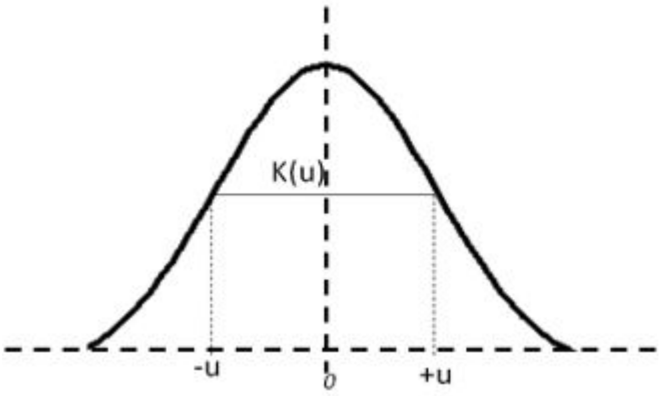


EVALUATING THE NETWORK

- To evaluate the performance of our network, we view deep networks through the lens of Kernel Regression.
- Recent ML theory work shows that training neural network with gradient descent becomes the same as performing kernel regression as the width of each hidden layer goes to infinity (the Neural Tangent Kernel)
- We want our Kernel to be Stationary & Shift-Invariant
- The Neural Tangent Kernel should be tunable to enable fast convergence.

THE KERNEL

1. A kernel function must be symmetrical. Mathematically this property can be expressed as $K(-u) = K(+u)$.
2. The symmetric property of kernel function enables its maximum value ($\max(K(u))$) to lie in the middle of the curve.
3. Value of kernel function **can not be negative i.e.** $K(u) \geq 0$ for all $-\infty < u < \infty$.



KERNEL REGRESSION

- Method for fitting a continuous function to a set of data points $\{(x_i, y_i)\}$
- High level: add up a set of kernel functions, one centered at each input point, each with its own weight
- Weights are optimal in a least-squares sense: $\min_w \sum_i \|y_i - \hat{f}_w(x_i)\|^2$

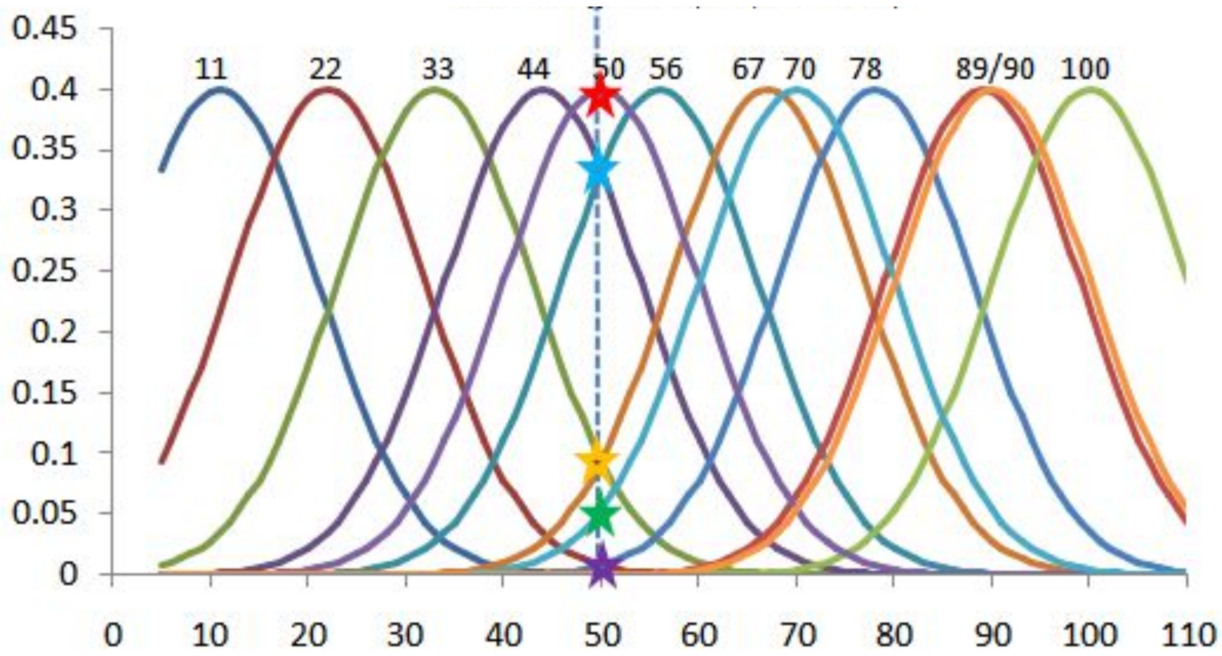
$$\hat{f}_w(x) = \sum_{i=1}^{n} w_i k(x - x_i)$$

Kernel centered at training input point x_i

Estimated function

Weight corresponding to kernel centered at x_i

KERNEL WIDTH



KERNEL WIDTH

- More width => far away points are also given considerable weights (Underfitting)
  Reconstruction is too smooth.

- Skinny kernel => only nearby points matter (Overfitting)
  Reconstruction is not interpolated correctly

- Tune the bandwidth appropriately.

NEURAL TANGENT KERNEL

The Neural Tangent Kernel for an MLP is a scalar function of dot product of two input vectors x and y

$$NTK(x,y) = h(x^T y)$$

Dot product of Fourier Feature Mapping :

$$\gamma(x)^T \gamma(y) = \sin(Bx)^T \sin(By) + \cos(Bx)^T \cos(By) = \cos(B(x-y))$$

Adding Fourier Feature changed our NTK to:

$$NTK(\gamma(x),\gamma(y)) = h(\cos(B(x-y))$$

$$\gamma(\mathbf{v}) = \left[a_1 \cos(2\pi \mathbf{b}_1^{\mathrm{T}} \mathbf{v}), a_1 \sin(2\pi \mathbf{b}_1^{\mathrm{T}} \mathbf{v}), \ldots, a_m \cos(2\pi \mathbf{b}_m^{\mathrm{T}} \mathbf{v}), a_m \sin(2\pi \mathbf{b}_m^{\mathrm{T}} \mathbf{v})\right]^{\mathrm{T}} .$$

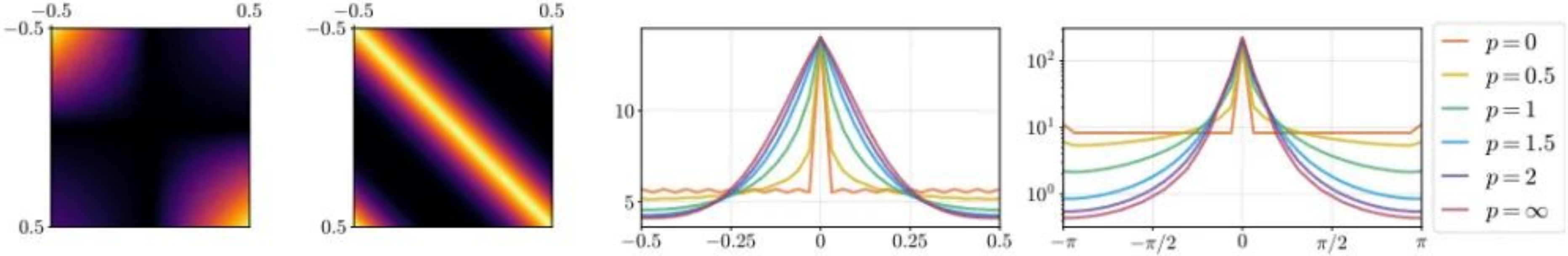
Because $\cos(\alpha - \beta) = \cos\alpha \cos\beta + \sin\alpha \sin\beta$, the kernel function induced by this mapping is

$$k_\gamma(\mathbf{v}_1, \mathbf{v}_2) = \gamma(\mathbf{v}_1)^{\mathrm{T}} \gamma(\mathbf{v}_2) = \sum_{j=1}^{m} a_j^2 \cos\left(2\pi \mathbf{b}_j^{\mathrm{T}} (\mathbf{v}_1 - \mathbf{v}_2)\right) = h_\gamma(\mathbf{v}_1 - \mathbf{v}_2)\,,$$

$$\text{where } h_\gamma(\mathbf{v}_\Delta) \triangleq \sum_{j=1}^{m} a_j^2 \cos(2\pi \mathbf{b}_j^{\mathrm{T}} \mathbf{v}_\Delta)\,.$$

- The Fourier feature mapping resulted in stationarity of the Neural Tangent Kernel.
- Sinusoidal input mapping transforms a dot product kernel into a stationary one, because we obtained a kernel function that is a difference of coordinates only.

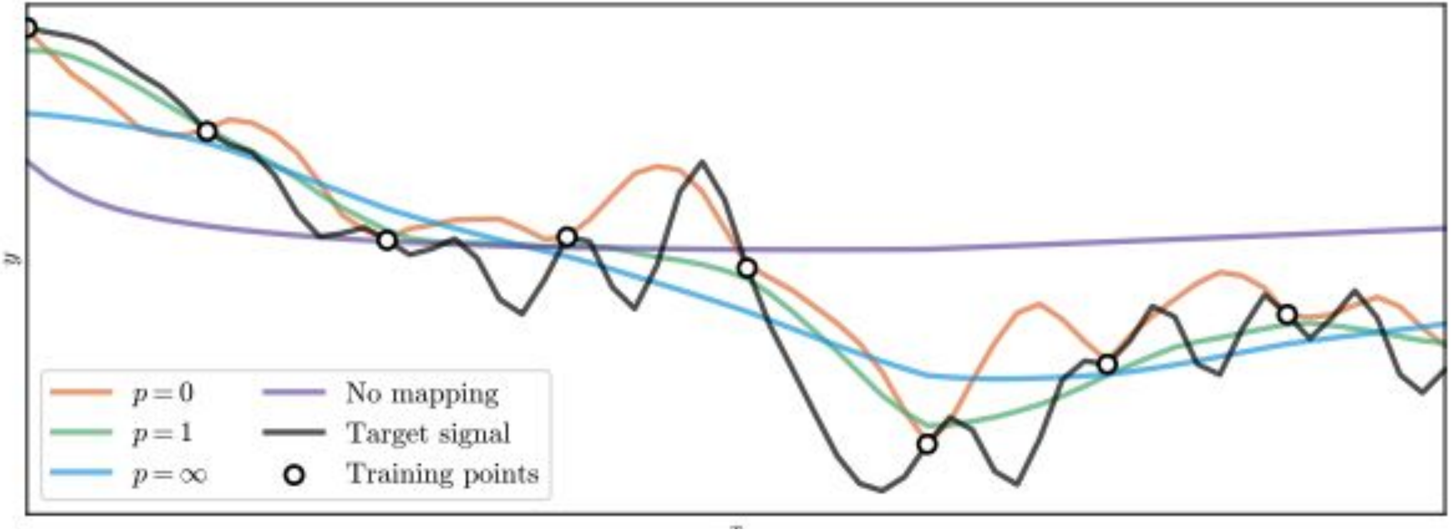
TUNING THE BANDWIDTH



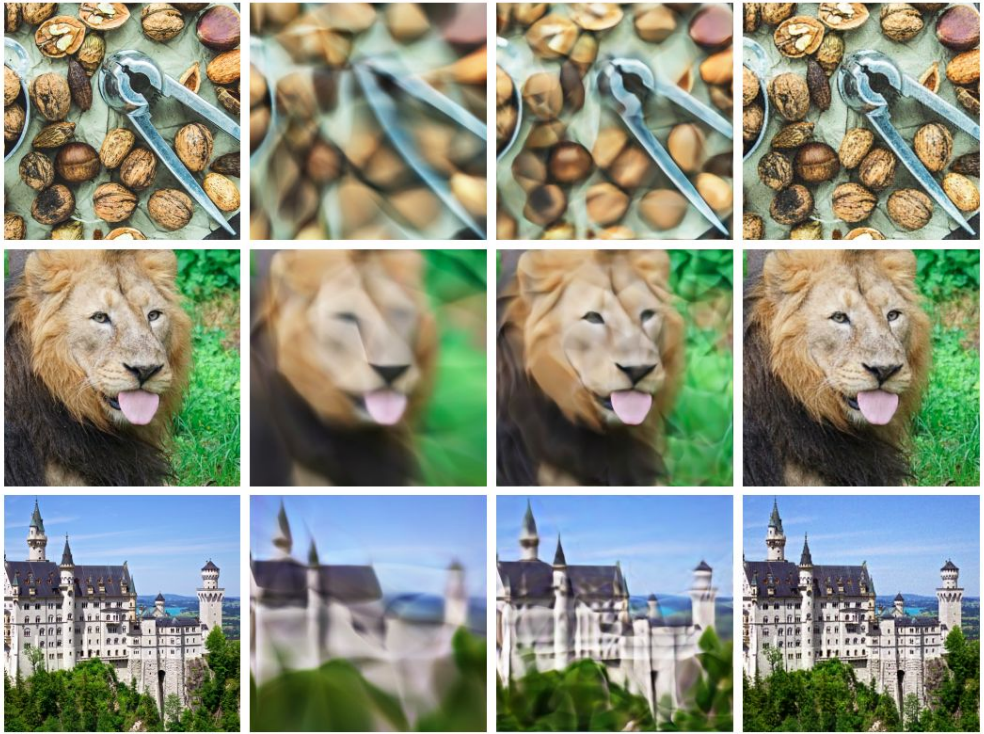
(a) No mapping NTK (b) Basic mapping NTK (c) NTK spatial (d) NTK Fourier spectrum

$$\gamma(\mathbf{v}) = \left[a_1 \cos(2\pi \mathbf{b}_1^{\mathrm{T}} \mathbf{v}), a_1 \sin(2\pi \mathbf{b}_1^{\mathrm{T}} \mathbf{v}), \ldots, a_m \cos(2\pi \mathbf{b}_m^{\mathrm{T}} \mathbf{v}), a_m \sin(2\pi \mathbf{b}_m^{\mathrm{T}} \mathbf{v})\right]^{\mathrm{T}} . \qquad (5)$$

To tune the bandwidth, different values are set for a_m and b_m and the corresponding NTK spectrum is observed.



It is shown that frequency mappings with lower p, resulted in wider spectra of the kernel, enabling faster convergence

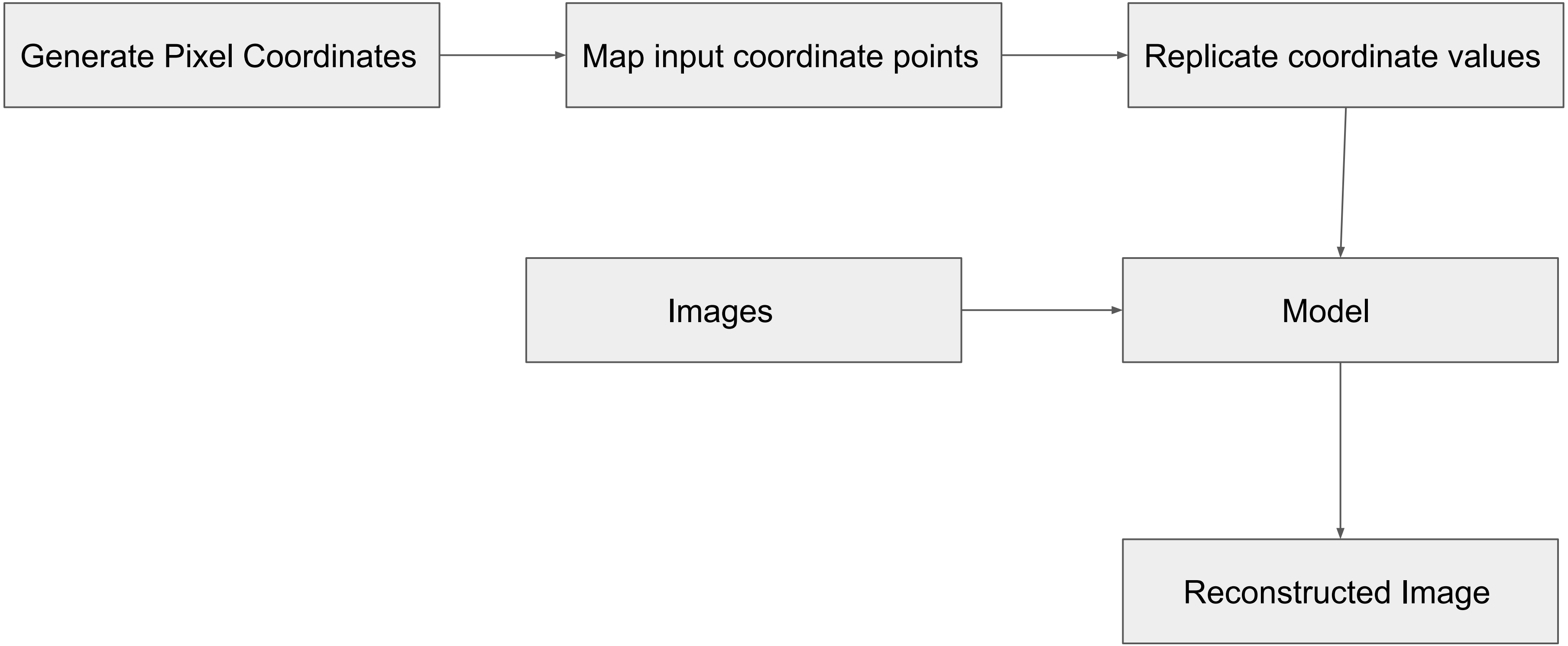


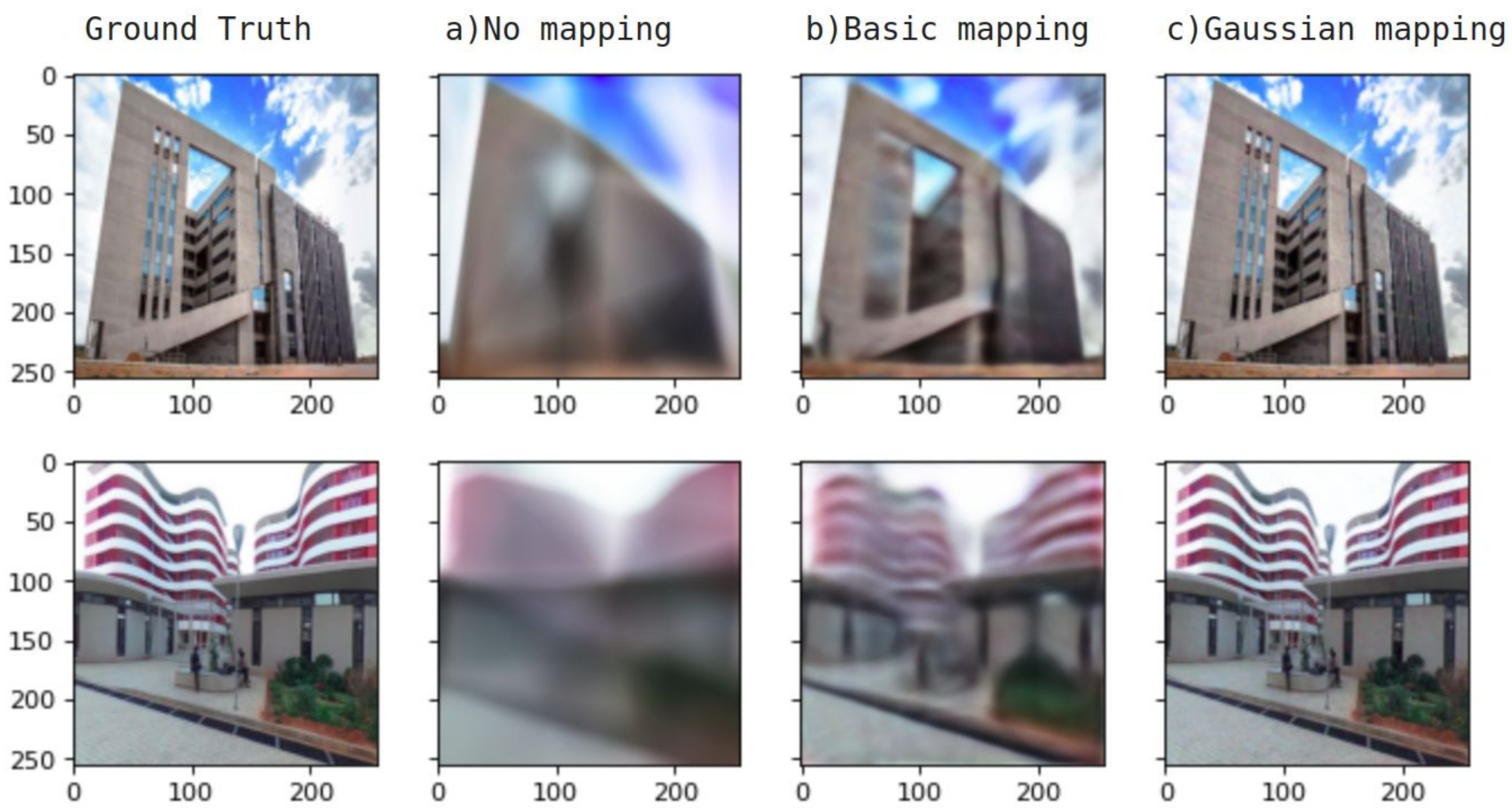
(a) Ground Truth (b) No mapping (c) Basic (d)Gaussian

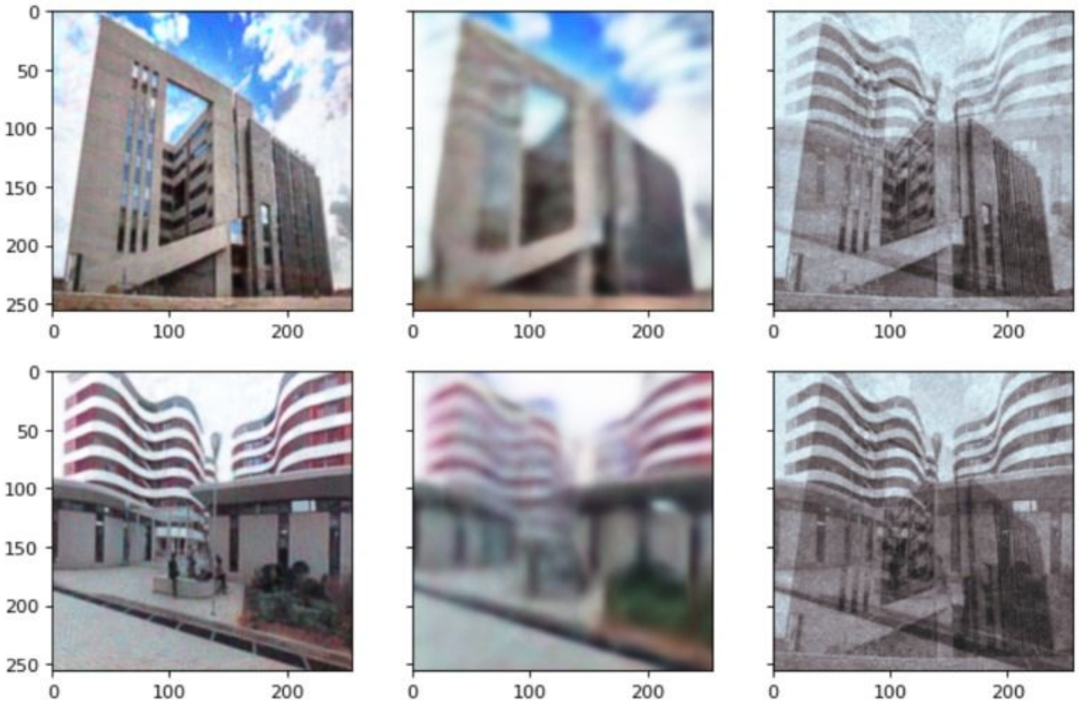
$\sigma = 1$ $\sigma = 2$ $\sigma = 10$ $\sigma = 32$ $\sigma = 64$

In experiments, the proposed Fourier feature mapping approach dramatically improves coordinate-based MLP performance across all tasks, with random Gaussian features performing best. The results illustrate the increasingly popular technique of using coordinate-based MLPs to represent 3D shapes in computer vision and graphics by using a simple mapping strategy.

IMPLEMENTATION




Ground Truth
a)No mapping
b)Basic mapping
c)Gaussian mapping
0
50
100
150
200
250
0
100
200



Sigma = 10

Sigma = 1
Underfitting

Sigma = 50
Overfitting