Supplementary Materials for: Around the World in 80 Timesteps: A Generative Approach to Global Visual Geolocation

Supplementary Material

In this appendix, we present our ablation study in Sec. A, and provide additional results and qualitative illustrations in Sec. B as well as further analysis C. We then provide implementation and technical details in Sec. D, and some technical elements and proofs in Sec. E.

A. Ablation Study

We conduct an ablation study on the Riemannian Flow Matching approach to evaluate the impact of our design choices, and report the results in Tab. A.

- **Guided Sampling.** Guided sampling improves the geoscore, but as shown in Figure 7 of the main paper, leads to low likelihood scores due to overconfident predictions.
- **Single sampling without guidance.** We do not add any guidance ($\omega = 0$ in Eq. 13). We observe a loss of geoscore of 182 GeoScore point (3485 vs 3767) , but the NLL is better (-1.8 vs 33.1). Guidance improves the geolocation performance but significantly worsen the probabilistic prediction.
- **Ensemble sampling.** We sample and denoise 32 random points and select the prediction with highest likelihood. While this approach yields the best performance for the distribution estimation metrics, it is significantly more computationally expensive due to the necessity of generating and evaluating multiple samples. In practice, this inflates the prediction time per image from approximately 2 milliseconds to 72 milliseconds..
- **Standard Sigmoid Scheduler.** We replace our proposed scheduler defined in Eq.15 of the main paper by the standard not skewed sigmoid scheduler with $\alpha = -3$ and $\beta = 3$. This modification increases the geoscore but decreases the quality of the predicted densities as measurs by the generative metrics. The standard sigmoid does not allocate sufficient emphasis to the earlier stages of the diffusion process (t close to 0: low noose regime), which are crucial for fine-grained localization.
- **Linear Sigmoid Scheduler.** We replace our proposed scheduler defined in Eq.15 of the main paper by a linear scheduler. This modification decreases both the geoscore and the quality of the predicted densities.

B. Additional Results

We present additional qualitative and quantitative results.

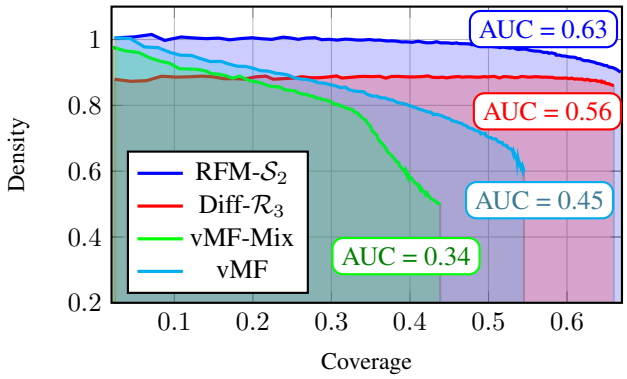


Figure A. **Density Coverage curves on OSV-5M.**

Qualitative Illustrations. We provide a detailed illustration of our network in Fig. B. We observe that the parametric methods vMF and vMF mixture fail to capture highly multimodal distributions. In contrast, our distributions are non-parametric and can predict highly complex spatial distributions. The vMF mixture is collapse to a single vMF, as we observed for a majority of the prediction.

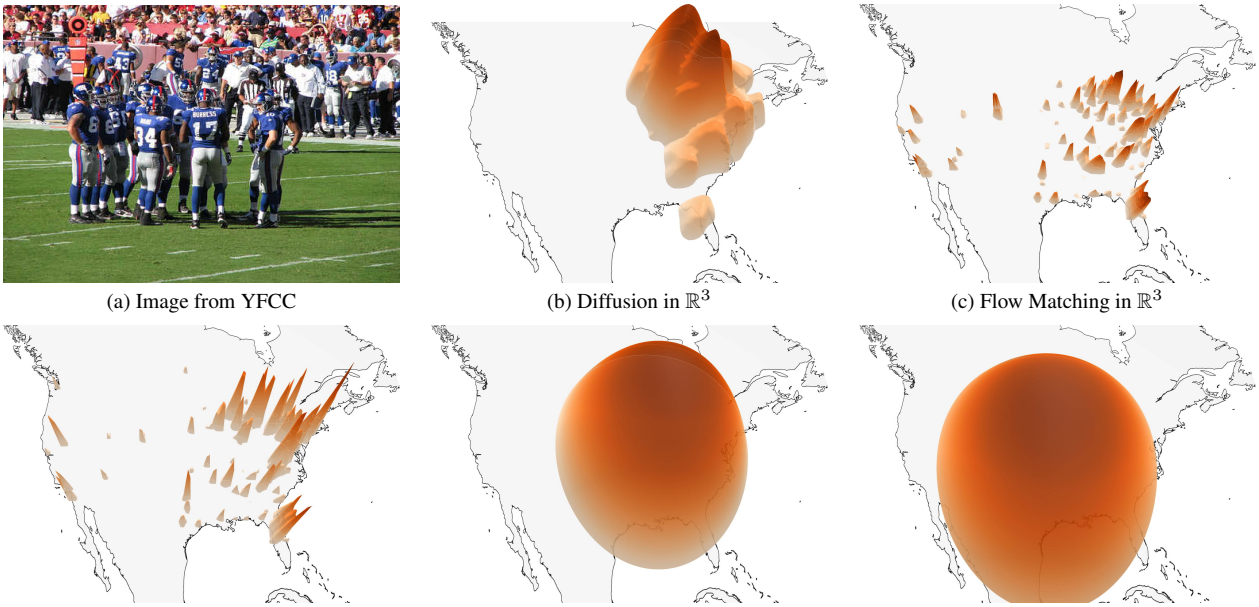
We observe that both flow matching approaches give results that visually close. Note however that the value of the likelihoods are not comparable as both models are not embedded in the same metric space. The generative metrics detailed in Tab. B show that the Riemannian model fits the unconditioned distribution better at a fine-grained scale.

Detailed Quantitative Results. We provide in Tab. B the full generative metrics for the OSV-5M and YFCC datasets. Similarly to what we observed for iNat21 in the main paper, flow matching and particularly Riemannian flow matching leads to the most faithful predicted distributions of samples.

C. Extended Analysis

We extend the analysis of our models' performance.

Speed. Iterative methods, such as flow matching, come with overhead compared to single-forward-pass baselines. However, our approach needs to encode the image only once. The iterative inference then operates solely on the image embedding and the coordinates, which is comparatively inexpensive. In practice, encoding an image takes about 20 ms, while each denoising step takes roughly 5 μs when averaged across 10,000 images. Even when using 16 steps, the



(a) Image from YFCC (b) Diffusion in $\mathbb{R}^3$ (c) Flow Matching in $\mathbb{R}^3$

(d) Riemannian Flow Matching in $\mathcal{S}_2$ (e) von Mises-Fisher (f) von Mises-Fisher Mixture

Figure B. **Qualitative Illustration.** We represent the predicted distributions predicted by different models for the same image, taken in an NFL stadium in Maryland, USA.

Table A. **Ablation Study.** We estimate the impact of different designs. We consider a Riemannian diffusion model and evaluate on OpenStreetView-5M.

	Geoscore ↑	NLL ↓	precision ↑	recall ↑	density ↑	coverage ↑
Guided sampling	3746.79	33.1	0.841	0.896	0.797	**0.590**
Single sampling	3485.88	-1.81	0.844	0.924	0.790	0.560
Ensemble sampling	3588.25	**-4.31**	**0.899**	0.785	**0.881**	0.537
Linear sigmoid	3734.84	-1.28	0.775	**0.931**	0.687	0.536
Standard sigmoid	**3767.21**	-1.51	0.827	0.913	0.765	0.565

overhead is only about 0.4%.

Interpreting Generative Metrics. Precision and recall in generative models can be confusing as they have different definitions and behaviors than their equivalent in a retrieval context. In a generative context, uniform baselines achieve high recall scores because they cover the entire support of the target distribution. To address this, previous work [4] suggests using density and coverage as more informative metrics. We propose to examine the density/coverage trade-off by sorting the predicted locations by decreasing likelihood and computing these metrics over the first entries in the list. The resulting curve is shown in Fig. A. As we can see, the AUC of our RFM model is well above other methods, maintaining a high density as the coverage increases.

Interpreting Localizability. The localizability score—expressed as an average negative entropy in bits per dimension [2, 7]—can be difficult to interpret in absolute terms. We primarily use it for comparative purposes across images or models. Nevertheless, we can offer three reference points:

- **Absolute Lower Bound.** A uniform distribution over the sphere sets a lower bound for localizability of $-\frac{1}{3}\log(4\pi) = -1.22$.
- **Empirical Lower Bound.** The unconditional distribution learned by our model has a localizability of -0.54, which is the lowest possible localizability our model can predict.
- **Empirical Upper Bound.** While there is no strict theoretical upper limit to the localizability, the image of the Eiffel tower in Fig 6c reached the highest observed localizability in our experiments with 1.75.

Table B. **Generative Metrics.** We evaluate the quality of the predicted distributions with generated metrics for OSV-5M and YFCC for the unconditional distribution.

	OSV-5M				YFCC			
	precision ↑	recall ↑	density ↑	coverage ↑	precision ↑	recall ↑	density ↑	coverage ↑
Uniform	0.29	0.98	0.21	0.21	0.59	0.99	0.38	0.22
vMF Regression	0.598	**0.982**	0.499	0.446	0.667	**0.993**	0.542	0.599
vMF Mixture	0.513	0.980	0.422	0.358	0.626	0.988	0.474	0.498
RFlowMatch $\mathcal{S}_2$ (ours)	0.841	0.896	0.797	**0.590**	**0.957**	0.952	**1.060**	**0.926**
Diffusion $\mathbb{R}^3$ (ours)	0.822	0.916	0.752	0.568	0.938	0.959	0.959	0.837
FlowMatch $\mathbb{R}^3$ (ours)	**0.845**	0.907	**0.799**	0.575	0.953	0.959	1.037	0.920

D. Implementation Details

Baseline Details. We use the same backbone and image encoder as in our model for all baselines. We adapt them to the baselines with two modifications: (i) The missing inputs (noisy coordinates and scheduler) are replaced by learnable parameters. (ii) We replace the final prediction head with MLPs that predict the parameters of the von Mises-Fisher (vMF) distribution: the mean direction $\mu \in \mathcal{S}^2$ (using L_2 normalization) and the concentration parameter $\kappa > 0$ (using a softplus activation).

For the mixture of vMF model, we use $K = 3$ vMF distributions. The μ and κ heads now predict three sets of parameters, and the mixture weights are predicted by another dedicated head (with a softmax activation).

Architecture Details. Our model architecture, illustrated in Fig. C, consists of several key components:

- **Input Processing:** The model takes three inputs: the current coordinate x_t, an image embedding c, and the noise level $\kappa(t)$.
- **Initial Transformation:** The coordinate x_t first passes through a linear layer that expands the dimension from 3 to d, followed by an ADA-LN layer that conditions on parameters α, β.
- **Main Processing Block:** The core of the network (shown in gray) is repeated N times and consists of:
  - A linear layer that expands dimension from d to $4d$
  - A GELU activation function
  - A linear layer that reduces dimension from $4d$ to d
  - An ADA-LN layer conditioned on α, β
- **AdaLN:** The AdaLN layer is a conditional layer normalization that scales and shifts the input based on the image features:

$$\text{AdaLN}(x) = \gamma \odot \frac{x - \mu}{\sigma} + \beta \tag{A}$$

  where μ, σ are the mean and standard deviation of x on the feature dimension, and γ, β are learnable parameters.
- **Skip Connections:** Each processing block has a skip connection path that:
  - Skips the processing block and directly connects the input to the output to allow a better gradient flow.
  - Is modulated by a gating parameter γ that controls how much of the block output is added to the main path.

  This gated skip connection allows the network to adaptively control information flow around each processing block.
- **Output Head:** The final prediction is obtained through a linear layer that maps to the target dimension $d \mapsto 3$.
- **Time step Conditioning:** The noise level $\kappa(t)$ is incorporated through addition to the conditioning of the AdaLN layers.

We use $N = 12$ blocks of dimension $d = 512$ for OSV-5M and YFCC-100M and blocks of dimension $d = 256$ for iNat21.

Optimization. We train our models for 1M steps with a batch size of 1024, using the Lamb optimizer [12] with a learning rate of $8 * 10^{-4}$. We use a warmup of 500 steps and a cosine decay learning rate schedule. We use an EMA of 0.999 for the model weights. For OSV-5M and YFCC-100M, we use a weight decay of 0.05 and for iNaturalist we use 0.1. We drop out 10% of the time the conditioning image embedding to allow classifier free guidance.

Metrics.

- **Precision and Recall:** We adapt the classic generation metrics of precision and recall [4] to our spatial setting by considering geographic proximity.
  We consider a set X of true locations, and a set Y of locations sampled from the unconditional distribution predicted by our model. For Z a set of locations (X or Y) and $z \in Z$, we define $\mathbf{B}(z, Z)$ the ball of $\mathcal{S}_2$ centered on z and with radius equal to the k-th nearest neighbour of z in Z. We can then define the approximated manifold of ta et of locations:

$$\text{manifold}(Z) := \bigcup_{z \in Z} \mathbf{B}(z, Z) \,. \tag{B}$$

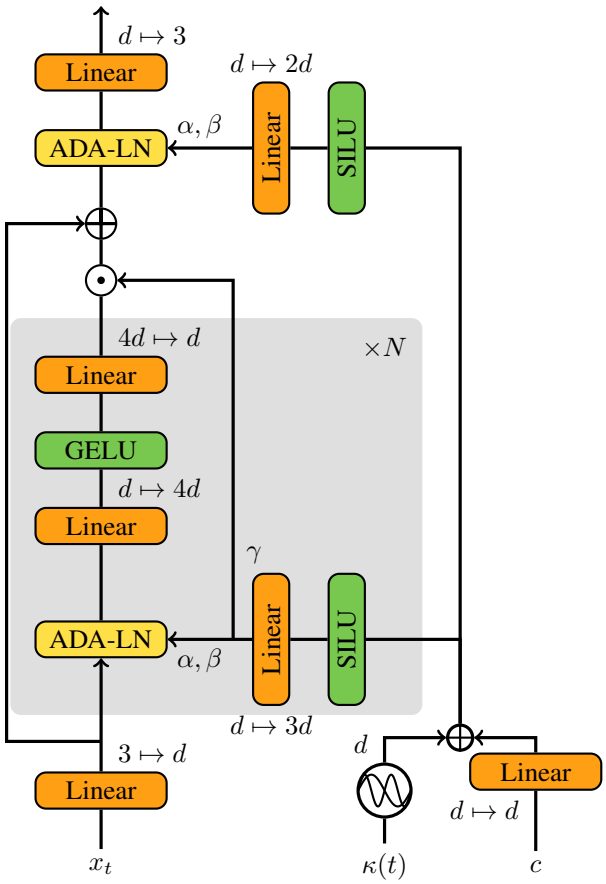


Figure C. **Architecture.** Our model takes as input the current coordinate x_t, the image embedding $\phi(c)$, and the noise level $\kappa(t)$. We use this architecture for all our formulations, including deterministic baselines.

We now define the precision and recall as the proportion of predicted (resp. true) locations within the manifold of true (resp. predicted) locations:

$$\text{precision} := \frac{1}{\mid Y \mid} \sum_{y \in Y} [y \in \text{manifold}(X)] \tag{C}$$

$$\text{recall} := \frac{1}{\mid X \mid} \sum_{x \in X} [x \in \text{manifold}(Y)] \, , \tag{D}$$

where $[P]$ is the Iverson bracket, equal to one is the statement P is true and 0 otherwise. Throughout this paper, we select the number of neighbours to $k = 3$.

- **Density and Coverage:** Naeem *et al*. [6] introduce more reliable versions of the precision and recall metrics, particularly for distributions containing outliers. We propose to adapt these metrics to our setting. The density measures how closely the predicted locations Y cluster around the true location X :

$$\text{density} := \frac{1}{k \mid Y \mid} \sum_{y \in Y} \sum_{x \in X} [y \in \mathbf{B}(x, X)] \, . \tag{E}$$

The recall metrics can be misleading high for predicted manifolds that cover uniformly the embeddings space, which is particularly problematic on a low- dimensional space such as $\mathcal{S}_2$: the uniform distribution has a recall of 0.98 on OSV-5M. Coverage better captures how well the generated distribution spans the true data modes without rewarding such overestimation by assessing how well the predicted distributions span the true data:

$$\text{coverage} := \frac{1}{\mid X \mid} \sum_{x \in X} [\exists y \in Y \cap \mathbf{B}(x, X)] \, . \tag{F}$$

E. Technical Details

In this section, we present details on Riemannian geometry on the sphere, and a proof sketch of Proposition 1 and elements on its generalization.

Spherical Geometry. The logarithmic map $\log_x$ maps a point $y \in \mathcal{S}_2$ onto T_x, the tangent space at point x [8]:

$$\log_x(y) = \frac{\theta}{\sin \theta}(y - \cos \theta x) \, , \tag{G}$$

where $\theta = \arccos(\langle x, y \rangle)$ is the angle between x and y. The exponential map $\exp_x$ of a point $x \in \mathcal{S}_2$ maps a tangent vector $v \in T_x$ back onto th sphere:

$$\exp_x(v) = \cos(\|v\|)x + \frac{\sin(\|v\|)}{\|v\|} v \, , \tag{H}$$

where $\|v\|$ is the Euclidean norm of v.

Proof of Prop 1. Please find here the corrected proposition and its proof. We now propose a short proof of Proposition 1, inspired by [5, Appendix C]

Proposition 1. *Given a location $y \in \mathcal{S}^2$ and an image c, consider solving the following ordinary differential equation system for t from 0 to 1:*

$$\frac{d}{dt} \begin{bmatrix} x(t) \\ f(t) \end{bmatrix} = \begin{bmatrix} \psi(x(t) \mid c) \\ -\operatorname{div} \psi(x(t) \mid c) \end{bmatrix} \text{ with } \begin{bmatrix} x(0) \\ f(0) \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix} \, , \tag{I}$$

Then the log-probability density of y given c is: $\log p(y \mid c) = \log p_\epsilon(x(1) \mid c) + f(1)$ where p_ϵ is the distribution of the noise ϵ, and $f(t)$ accumulates the divergence of the velocity field along the trajectory.

Proof. The logarithmic mass conservation theorem [1, 11] writes:

$$\frac{d}{dt}\log p(x_t \mid c) + \text{div } v(x_t) = 0\,. \tag{J}$$

After training the network ψ to regress $v(x_t)$, we can substitute $\psi(x_t \mid c)$ to $v(x_t)$ and obtain:

$$\frac{d}{dt}\log p(x(t) \mid c) + \text{div } \psi(x(t) \mid c) = 0\,. \tag{K}$$

We integrate from 0 to 1:

$$\log p(x_1 \mid c) - \log p(x(0) \mid c) = -\int_0^1 \text{div } \psi(x(t) \mid c)\,. \tag{L}$$

We thus have the following system:

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ f(t) \end{bmatrix} = \begin{bmatrix} \psi(x(t) \mid c) \\ -\text{div } \psi(x(t) \mid c) \end{bmatrix} \tag{M}$$

with initial condition:

$$\begin{bmatrix} x(0) \\ f(0) \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}\,. \tag{N}$$

Where accumulates the divergence of the velocity field along the trajectory: $f(t) = \int_0^t \text{div}\,\psi(x(t) \mid c)$ and hence $f(0) = 0$. The system in Eq. (M) admits only one solution for all $t \in [0, 1]$. Equation (O) gives us that:

$$\log p(x_0 \mid c) = \log p(x(1) \mid c) - f(1)\,. \tag{O}$$

The probability $\log p(x(1) \mid c)$ is given directly by the distribution of the initial noise,a nd $f(1)$ is the solution of the system for f at $t = 1$.

□

Extending Prop 1. Prop 1 can be extended to Riemenian Flow Matching simply by projecting the iterate onto the sphere at each step when iteratively solving the ODE Eq. (M).
For diffusion models, we do not have direct access to the velocity field. However, according to Song *et al.* [10, Section D.2], for a stochastic differential equation of the form:

$$dx = f(x,t)dt + G(x,t)d\omega \tag{P}$$

where $d\omega$ is a Wiener process [3], the velocity field $\Psi(x,t)$ can be expressed as:

$$v(x,t) = f(x,t) - \frac{1}{2}\nabla \cdot [G(x,t)G(x,t)^T] - \frac{1}{2}G(x,t)G(x,t)^T \nabla \log p_t(x_t \mid x_0, c) \tag{Q}$$

In our case, we defined our forward noising process as:

$$x_t = \sqrt{1-\kappa(t)}x_0 + \sqrt{\kappa(t)}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)\,. \tag{R}$$

This leads us to choose:

$$f(x,t) = -\frac{1}{2}x\beta(t) \tag{S}$$

$$G(x,t) = \sqrt{\beta(t)}\,, \tag{T}$$

where $\beta(t)$ represents the infinitesimal change in x_t variation between t and $t + \delta t$: $\beta(t) = x_{t+\delta t} - x_t$. According to [10, Eq 29], this process yields:

$$x_t \sim \mathcal{N}\left(x_0 e^{-\frac{1}{2}\int_0^t \beta(s)ds}, \left(1 - e^{-\int_0^t \beta(s)ds}\right) I\right) \tag{U}$$

which implies that [9, X]:

$$\beta(t) = \frac{d\log(\kappa(t))}{dt} \tag{V}$$

Finally, we can replace $\nabla \log p_t(x_t \mid x_0, c)$ with $-\epsilon_\theta(x_t, t, c)$ in Eq. (Q), as our model learns to predict the noise added to the data. This yields the following velocity field:

$$\psi(x,t) = -\frac{1}{2}\beta(t)(x - \epsilon_\theta(x,t,c))\,. \tag{W}$$

References

[1] Ben-Hamu, H., Cohen, S., Bose, J., Amos, B., Nickel, M., Grover, A., Chen, R.T., Lipman, Y.: Matching normalizing flows and probability paths on manifolds. In: ICML (2022) 5

[2] Chen, R.T., Lipman, Y.: Riemannian flow matching on general geometries. In: ICLR (2024) 2

[3] Durrett, R., Durrett, R.: Probability: Theory and examples. Cambridge university press (2019) 5

[4] Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved precision and recall metric for assessing generative models. NeurIPS (2019) 2, 3

[5] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: ICLR (2023) 4

[6] Naeem, M.F., Oh, S.J., Uh, Y., Choi, Y., Yoo, J.: Reliable fidelity and diversity metrics for generative models. In: ICML (2020) 4

[7] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022) 2

[8] Sommer, S., Fletcher, T., Pennec, X.: Introduction to differential and riemannian geometry. In: Riemannian Geometric Statistics in Medical Image Analysis. Elsevier (2020) 4

[9] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: ICLR (2021) 5

[10] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: ICLR (2021) 5

[11] Villani, C.: Optimal transport: Old and new. Berlin: Springer (2009) 5

[12] You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., Hsieh, C.J.: Large batch optimization for deep learning: Training bert in 76 minutes. ICLR (2020) 3