# A digital twin approach to surgery planning combining differentiable simulations to Virtual Reality: Supplementary Information


## ABSTRACT

This supplementary document provides additional details and examples to complement our main article. We begin by outlining the initialization methods for the navigation field optimization process. Next, we elaborate on the components of the loss function used in our differentiable simulation. To illustrate the versatility of our proposed navigation algorithms, we present their application across various medical cases. We then showcase different representations of the sampled solutions, offering insights into the algorithm's outputs. Additionally, we provide a comprehensive list of hyper-parameters used in the differentiable simulations discussed in the main paper. Lastly, we briefly introduce the napary plugin, a tool we developed for medical image visualization, which enhances the interpretability of our results.


## S1 Navigation field initialization method

We provide elements of the various initialisation schemes used to run our applications.

### S1.1 Artificial potential field

The Artificial Potential Field (APF) method[1] is based on creating a potential field around the agent and obstacles within its environment. The potential field influences the agent's movement towards the target while avoiding collisions with obstacles. The attractive potential, $U_{\text{att}}(\mathbf{r})$, reads

$$U_{\text{att}}(\mathbf{r}) = \frac{1}{2} k_{\text{att}} \|\mathbf{r} - \mathbf{r}_g\|^2$$

where $\mathbf{r}$ represents the position of the agent and $\mathbf{r_g}$ is the position of the target. The repulsive potential, $U_{rep}(\mathbf{r})$, reads :

$$U^o_{\text{rep}}(\mathbf{r}) = \begin{cases} \frac{1}{2} k_{\text{rep}} \left( \frac{1}{\rho(\mathbf{r})} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho(\mathbf{r}) \leq \rho_0 \\ 0 & \text{if } \rho(\mathbf{r}) > \rho_0 \end{cases}$$

where $\rho(\mathbf{r})$ is the minimum distance from the agent to the closest point on the obstacle, $\rho_0$ is the influence distance of the obstacle, and $k_{\text{rep}}$ is the scaling factor for the repulsive field. The total potential field reads

$$U_{\text{tot}}(\mathbf{r}) = U_{\text{att}}(\mathbf{r}) + \sum_o U_{\text{rep}}(\mathbf{r})$$

Therefore, $U_{\text{tot}}(\mathbf{r})$ can be used to determine an initial value of the velocity field. Limitations of the procedure are the multiple local minima of the spatial field and challenges in optimising the hyperparameters, especially if multiple spatial scales are present in the obstacles field.

### S1.2 Harmonic navigation function

We have investigated the use of harmonic fields to establish an initial velocity field. Harmonic functions, which are solutions to Laplace's equation, $\Delta f = 0$, where $\Delta f$ represents the Laplacian of function $f$, possess a key property known as the "maximum principle". This principle states that if $f$ is defined on a compact set $K$, it attains its maximum and minimum values on the boundary of $K$. Consequently, a harmonic function cannot exhibit multiple local minima or maxima, a challenge commonly associated with potential field approaches. By enforcing boundary conditions at the goal and obstacles, Connolly et al.[2] demonstrated that an agent governed by a harmonic potential field will not be trapped in local potential minima, as such points do not exist within the free space of the environment.

Furthermore, since every streamline of a harmonic function must culminate at a goal point, an agent following the gradient of a harmonic potential is ensured to reach the goal, provided it is within the same contiguous free space. The absence of local minima and the guarantee of goal progression significantly enhance traditional Artificial Potential Field (APF) methods.

Additionally, Kim et al.[3] proved that harmonic functions are robust in environmental model uncertainties.

We used a classical implementation[4]: we approximated the Laplacian operator using finite difference method and we imposed Dirichlet boundary conditions at the target position and on the segmentation.

### S1.3 Coupling A* solution with Laplace equation

In the case of large domain simulations, harmonic function and Dirichlet boundary conditions lead to inefficient implementation. For large domains, we favoured the use of A* algorithm[5]. It is an efficient pathfinding and graph traversal method. It finds the shortest path between a start node and a goal node in a weighted graph, where each edge has an associated cost.

Once a valid path is found on the graph whose edges are the discretization cells and the vertices are neighbouring cells, one can assign a high value to the initial position and a low value to the target and solve the 1D Laplace equation. The obtained solution can then serve as a new set of Dirichlet boundary conditions for the 3D Laplace equation problem where cells of the path returned by A* are now part of the domain's boundary. Figures 5(a) and 5(g) illustrate this initialisation method. These figures show that the A* solution follows the domain's frontier, leading to trajectories too close to the trachea. Then, optimization is required to avoid colliding with the domain's frontier.

## S2 Loss function estimation

We detail all the terms of the loss function $L(\mathbf{P},\mathbf{v})$ that is being optimized during stochastic gradient descent (SGD)[6]. The key requirement here is that all the loss functions are differentiable with respect to the velocity vector field $\mathbf{v}$.

Let $\mathbf{r_n^t}$ be the position of the $n$-th random walker of the batch at simulation time $t$, $\mathbf{r_g}$ the position of the target, and $\mathbf{v}\left(\mathbf{r}\right)$ the spatially dependent vector field in the domain. Let $N$ be the total number of agents in an batch and $T$ be the differentiable simulation total duration.

The target loss, which promotes reaching the target, can be defined by the the L2 norm between the position of each agent in a batch and the target goal position. It reads:

$$L_{target}(\mathbf{P},\mathbf{v}) = \frac{1}{N}\sum_{n=1}^{N}||\mathbf{r_n^T}-\mathbf{r_g}||^{\mathbf{2}} \tag{S1}$$

As the position at each time step is obtained by Euler-Maruyama integration of the Langevin equation (1), given by the following equation, the final position of each agent is a differentiable function of the vector field.

$$\mathbf{r_n^{t+1}} = \mathbf{r_n^t} + \delta\mathbf{t}\mathbf{v}(\mathbf{r_n^t}) + \sqrt{\mathbf{2D}\delta\mathbf{t}}\varepsilon$$

where $\delta t$ is the differentiable simulation time-step, $D$ is the diffusion coefficient and $\varepsilon \sim \mathscr{N}(0,1)$ is a white noise added as a perturbation term that is omitted during back-propagation of the gradient with respect to $\mathbf{v}$.

Let also denote $\theta_{t,n}$ the angle, measured in radians, between $\mathbf{v}\left(\mathbf{r_n^t}\right)$ and $\mathbf{v}\left(\mathbf{r_n^{t+1}}\right)$. A differentiable estimation of the path bending constraint using the the local vector field can be expressed as the squared angle $\theta_{t,n}^2$ between two successive steps. Therefore, the bending loss, which encodes the physical constrains along the path of the surgical instrument, reads:

$$L_{smoothness}(\mathbf{P},\mathbf{v}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\theta_{i,n}^2 \tag{S2}$$

In practice, the angle $\theta_{t,n}$ is estimated using the cosine similarity between the vectors $\mathbf{v}\left(\mathbf{r_n^t}\right)$ and $\mathbf{v}\left(\mathbf{r_n^{t+1}}\right)$. The corresponding formulation reads:

$$\theta_{t,n} = arccos\frac{<\mathbf{v}\left(\mathbf{r_n^t}\right),\mathbf{v}\left(\mathbf{r_n^{t+1}}\right)>}{||\mathbf{v}\left(\mathbf{r_n^t}\right)||\mathbf{v}\left(\mathbf{r_n^{t+1}}\right)||}$$

Finally, lets denote $d_{\text{obs}}(\mathbf{r})$ the signed distance function of the obstacle map at position $\mathbf{r}$ and $\phi_{t,n}$ the angle between the vector field evaluated at position $\mathbf{r_n^t}$, $\mathbf{v}\left(\mathbf{r_n^t}\right)$ and $\nabla d_{\text{obs}}(\mathbf{r_n^t})$ which is the spatial gradient of the previously defined signed distance function. The angle $\phi_{t,n}$ is calculated in the same way as $\theta_{t,n}$.

The loss function associated with segmentation uncertainty, which attempts to maintain large distances from obstacles, can be written as a weighted sum of angle difference $\phi_{t,n}$ between the vector field and the signed distance map gradient at a given position $\mathbf{r_n^t}$ with a weighting coefficient corresponding to the relative distance to the obstacles:

$$L_{obstacle}(\mathbf{P},\mathbf{v}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=0}^{T-1}\phi_{i,n}^{2}\exp\left(-\frac{1}{2}\frac{d_{\text{obs}}^{2}(\mathbf{r_n^t})}{l_{min}^{2}}\right) \tag{S3}$$

where $l_{\text{min}}$ is a minimum collision length set by the user.

The total loss is a linear combination of the previously defined loss functions :

$$L(\mathbf{P},\mathbf{v}) = w_1 L_{target}(\mathbf{P},\mathbf{v}) + w_2 L_{obstacle}(\mathbf{P},\mathbf{v}) + w_3 L_{smoothness}(\mathbf{P},\mathbf{v}) \tag{S4}$$

where $(w_1, w_2, w_3)$ are the hyper parameters of the models.

## S3 Parametric exploration on the lung dataset

The parametric exploration of the effect of diffusivity parameter $D$ (figure 4 in the main text) was performed with 2000 walkers/paths/trajectories and 2000 time steps per simulation. In problem 1 (panels (a) and (b)), optimization trials were 200 epochs long and $D_0$ was set constant *wrt* the optimization epochs. In problem 2 (panels (c) and (d)), $D_0$ linearly decreased from the reported value in (d) down to the inferior value in the search grid (*e.g.* from 5 to 2), over 500 optimization epochs.

In addition, the local diffusivity $D_{\mathbf{r}}$ was modulated by the distance to the target:

$$D_{\mathbf{r}} = D_0 \frac{\|\mathbf{r}-\mathbf{r_g}\|}{\|\mathbf{r^{(t=0)}}-\mathbf{r_g}\|}$$

The above adjustment was performed to make the walkers stick to the target once they have reached it, independently of any cost function. This was used in combination with a modified distance cost, based on the nearest position of a walker to the target (instead of the last position at $t = T$):

$$L_{distance} = \frac{1}{N}\sum_{n=1}^{N}\min\left(1,\ \min_{t}\frac{\|\mathbf{r_n^t}-\mathbf{r_g}\|^2}{\|\mathbf{r_n^0}-\mathbf{r_g}\|^2}\right)$$

The above distance cost was found to largely accelerate the optimization in the absence of initialisation (*i.e.* starting with flat/null vector field). All the generated walker positions can drive the optimization instead of the terminal point only, which increases the probability a walker gets informed by the target location. On the other side, the distance is normalised and thresholded so that the loss is not dominated by "lost" walkers. As a downside, this cost function does not penalize the walkers that leave the neighbourhood of the target after they have reached it, hence the need for locally adjusting the diffusion.

The other cost functions were omitted ($w_2 = w_3 = 0$) to draw a simpler picture of the behaviour of the optimization procedure.

Another tweak was introduced to prevent the noisy walkers to "jump" over spatial regions outside the navigation domain, for example from a bronchial tube to another, in presence of high diffusion. This tweak consists in truncating the distribution of the norm of the random displacement based on a local measurement of the domain "width". The Euclidean Distance Transform was used to search for — at each voxel $\mathbf{r}$ in the domain — the nearest voxel $\mathbf{r_{edt}}$ outside the domain, and the nearest voxel outside the domain in the opposite direction (along the axis $\mathbf{r}-\mathbf{r_{edt}}$). The distance between these two opposite voxels was taken as a measurement of size of the local section of the domain (or diameter), and half this distance was used as a threshold on the random component of the displacements from $\mathbf{r}$.

The velocity vectors had their norm clipped at 1 (upper bound).

The source code of this particular parametric exploration is available at `https://github.com/francoislaurent/napari-potential-field-navigation/blob/diffusivity-sweep/src/napari_potential_field_navigation/diffusivity_sweep.py`.

## S4 Probabilistic path representation methods

While trajectories are generated with a stochastic component to ensure proper path explorations and robustness of the optimization, we can provide the user with various outputs to guide the path planning procedure to be visualised in VR. We provide, here, two examples, of possible outputs with

- **Temporal Mean of Sampled Trajectories**: This method involves sampling multiple trajectories and computing their temporal mean, providing an averaged representation of the potential paths. This approach offers a straightforward visualization of the most likely route while accounting for variability.

- **Kernel-Based Probability Estimation**: This approach utilizes kernel-based estimation techniques applied to sampled trajectories. By employing kernel density estimation, this method allows a representation of path probabilities, capturing the underlying distribution of potential routes. This technique effectively represents the probability landscape, especially in scenarios with complex or multimodal distributions of possible paths.

Figure S1 illustrates these two probabilistic path representations, providing a visual comparison of their respective outputs and highlighting the differences in their approaches to capturing path uncertainty.

(a) (b)

**Figure S1.** Examples of probabilistic path representations. (a) Left: Sampled paths (orange lines) and their temporal average (bold red line). This method provides a visualization of the most likely route while accounting for variability in the sampled trajectories. Note that in complex geometries, the temporal paths may lead to trajectories crossing boundaries (b) Right: Probability density estimated using kernel methods. The color gradient represents the probability of a path being taken, with warmer colors indicating higher probabilities. This approach does not lead to high probability trajectories crossing domain boundaries

## S5 Experimental evaluation

We show examples of applications of our approach with:

- **Lung Examples**: We leveraged additional lung images from the ATM22 Challenge[7] dataset, which focuses on airway segmentation from CT scans. This dataset is vital for analyzing pulmonary diseases and planning bronchoscopy-assisted surgeries. The images presented a range of lung morphologies and pathologies, enabling us to evaluate our method's performance across various lung-related path-planning scenarios.

- **Liver Navigation**: To assess the adaptability of our approach, we applied it to liver navigation tasks using images from the 3D-IRCADb-01[8] liver segmentation dataset. This dataset includes CT scans from 20 patients (10 women, 10 men), with 75% of the cases exhibiting hepatic tumors along with their corresponding segmentations. It offers a diverse set of liver morphologies, tumor locations, and segmentation challenges, making it well-suited for evaluating liver navigation algorithms.

The application of our method to these diverse datasets is illustrated in Figures S2 and S3. Figure S2 presents examples of lung navigation using CT scans from the airway segmentation dataset. Figure S3 showcases the liver navigation scenario using an example from the 3D-IRCADb-01 database, highlighting the method's adaptability to different anatomical contexts and its ability to plan paths around hepatic tumors and other obstacles. These figures visually represent the versatility of our approach across different medical imaging applications.

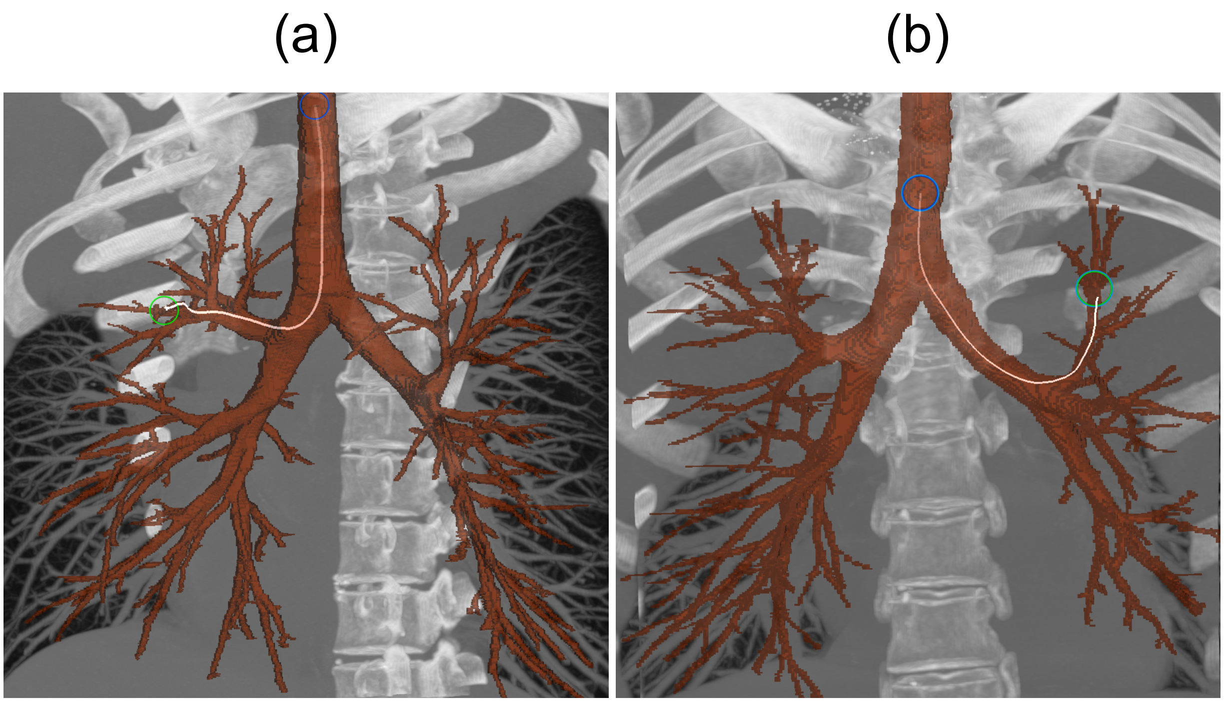


**Figure S2.** Lung navigation scenarios using two different samples of the ATM22 Challenge dataset. This example illustrates the method's capacity to adapt to pathological lung conditions while maintaining safe and efficient route planning.

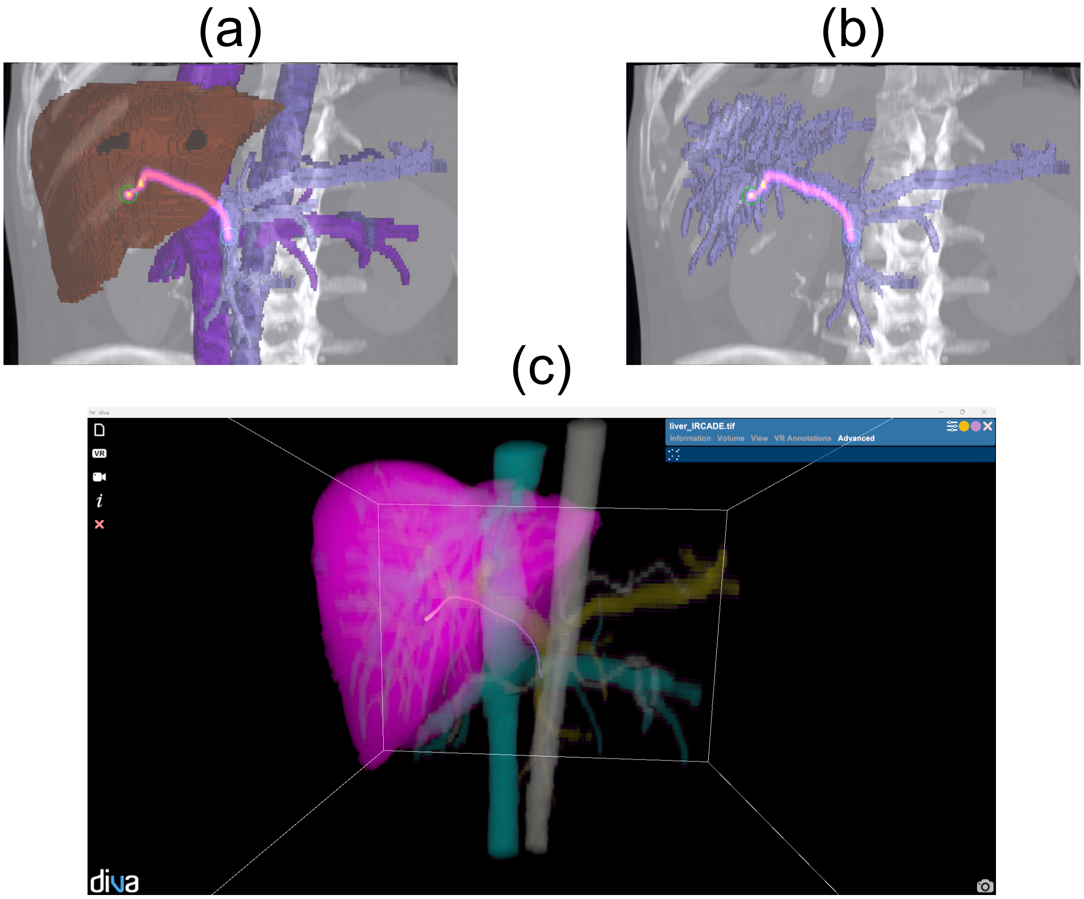


**Figure S3.** Liver navigation scenario using the 3D-IRCADb-01 dataset. (a) Top left panel: Complete liver segmentation with anatomical structures color-coded as follows: liver parenchyma (brown), portal vein (light blue), hepatic venous system (light purple), and hepatic artery (purple). The probability density of the navigation path is overlaid using warmer colors indicating higher probabilities. (b) Top right panel: The same view as (a), but with only the portal vein visible, allowing for a clearer visualization of the path density in relation to the vascular structure. This representation highlights the algorithm's ability to plan paths that avoid critical vascular structures. (c) Bottom panel: A screenshot from the DIVA software, demonstrating the potential for interactive, virtual reality-based exploration of the liver model and planned navigation paths. This VR interface enables surgeons and clinicians to intuitively interact with and evaluate the proposed navigation routes in a three-dimensional space.

## S6 Differentiable simulations hyper-parameters

We provide all the parameters of the 2D and 3D differentiable simulations presented in the paper. The target switch example (fig 3a, fig 4a) is identified as *switch*, the bending constraint optimization (fig 3c, fig 3d) by *bending* and the segmentation uncertainty avoidance example (fig 3e, fig 4g) by *uncertainty*. When obstacle loss is present in the total loss, the collision length $l_m in$ of the obstacle loss function equals 40.

| name | $w_{dist}$ | $w_{bend}$ | $w_{obs}$ | $T_{max}$ | dt | $N_{agent}$ | diffusivity | $N_{iter}$ | lr |
|---|---|---|---|---|---|---|---|---|---|
| switch | 1.0 | 0.0 | 0.0 | 50 | 0.1 | 100 | 0.2 | 200 | 0.1 |
| bending | 1.0 | 10.0 | 0.0 | 50 | 0.1 | 20 | 0.1 | 200 | 0.01 |
| uncertainty | 100.0 | 0.0 | 1.0 | 100 | 0.1 | 100 | 0.1 | 200 | 0.01 |

**Table S1.** Hyper-parameter values for 2D study case

| name | $w_{dist}$ | $w_{bend}$ | $w_{obs}$ | $T_{max}$ | dt | $N_{agent}$ | diffusivity | $N_{iter}$ | lr |
|---|---|---|---|---|---|---|---|---|---|
| switch | 1.0 | 0.0 | 0.0 | 2000 | 1.0 | 50 | 0.3 | 500 | 0.05 |
| bending | 1.0 | 10.0 | 0.0 | 200 | 1.0 | 100 | 0.01 | 200 | 0.01 |
| uncertainty | 1.0 | 0.0 | 10.0 | 1500 | 1.0 | 100 | 0.1 | 100 | 0.1 |

**Table S2.** Hyper-parameter values for 3D study case

## S7 Napari plugin overview


**Figure S4.** Screenshot of the Napari plugin interface

We present a brief overview of the Napari plugin developed to interact with 3D medical images. Napari is a Python library for $n$-dimensional image visualisation, annotation, and analysis. Existing plugins allow us to work with ITK for medical image visualization and annotation. Its minimalist interface makes it very intuitive to use and its 3D rendering tool makes medical data visualisation and interaction easy, as shown in figure S4.
The figure S5 is a screenshot of the trajectory visualisation plugin inside the DIVA[9] software.

The developed plugin contains:

- an I/O interface to load medical images and its associated label map,
- editing tools to modify the label map on the fly,
- selection of initial and goal point for path planning initialisation,
- visualisation of the initial potential field and its associated vector field,
- physical simulation and optimization parameters tuning,
- trajectories and mean trajectories visualisation,
- trajectories export in a CSV file that can be loaded by DIVA.

A beta version of the software can be downloaded on Napari Hub.

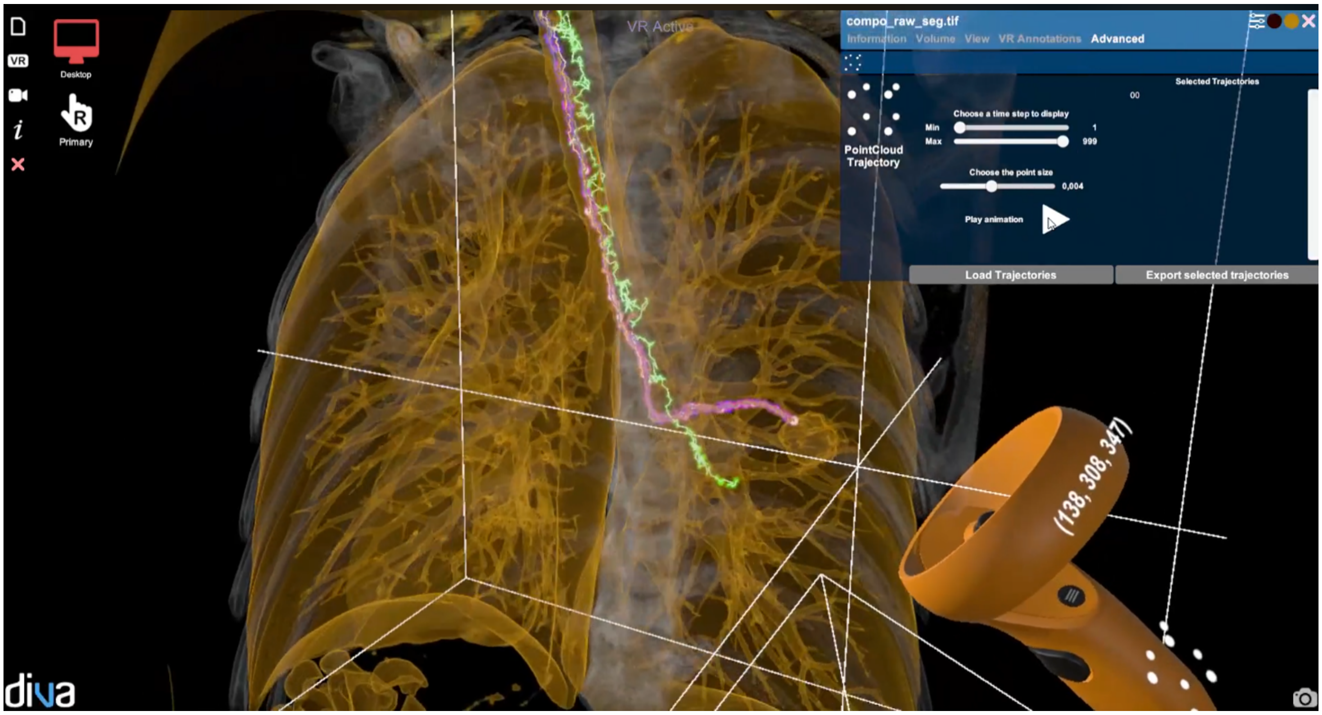


**Figure S5.** Screenshot of the VR navigation interface. A walkthrough video of the software is provided in supplementary

## S8 Hardware details

All development and testing were performed on a single platform equipped with an Intel(R) Xeon(R) W-1350 processor (3.30GHz, 6 cores, 12 logical processors), 16GB of RAM, and an NVIDIA GeForce GTX 1080 graphics card with 8GB of VRAM.

The DIVA software was tested using an Oculus Rift S headset. While DIVA software is compatible with previous generations of Oculus devices, forward compatibility with newer Meta/Oculus products has not been tested.

## References


**1.** Khatib, O. The Potential Field Approach And Operational Space Formulation In Robot Control. In Narendra, K. S. (ed.) *Adaptive and Learning Systems: Theory and Applications*, 367–377, DOI: 10.1007/978-1-4757-1895-9_26 (Springer US, Boston, MA, 1986).

**2.** Connolly, C., Burns, J. & Weiss, R. Path Planning Using Laplace's Equation. In *, IEEE International Conference on Robotics and Automation Proceedings*, 2102–2106 vol.3, DOI: 10.1109/ROBOT.1990.126315 (1990).

**3.** Kim, J.-O. & Khosla, P. Real-Time Obstacle Avoidance Using Harmonic Potential Functions. In *1991 IEEE International Conference on Robotics and Automation Proceedings*, 790–796 vol.1, DOI: 10.1109/ROBOT.1991.131683 (1991).

**4.** Connolly, C. I. & Grupen, R. A. The Applications of Harmonic Functions to Robotics. *J. Robotic Syst.* **10**, 931–946, DOI: 10.1002/rob.4620100704 (1993).

**5.** Hart, P. E., Nilsson, N. J. & Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Syst. Sci. Cybern.* **4**, 100–107, DOI: 10.1109/TSSC.1968.300136 (1968). Conference Name: IEEE Transactions on Systems Science and Cybernetics.

**6.** Bishop, C. M. *Pattern Recognition and Machine Learning* (Springer, New York, NY, 2016), softcover reprint of the original 1st ed. 2006 edition edn.

**7.** Zhang, M. *et al.* Multi-site, multi-domain airway tree modeling. *Med. image analysis* **90**, 102957 (2023).

**8.** Soler, L. *et al.* 3d image reconstruction for comparison of algorithm database. *URL: https://www. ircad. fr/research/data-sets/liver-segmentation-3d-ircadb-01* (2010).

**9.** El Beheiry, M. *et al.* DIVA: Natural Navigation Inside 3D Images Using Virtual Reality. *J. Mol. Biol.* **432**, 4745–4749, DOI: 10.1016/j.jmb.2020.05.026 (2020).