Supplementary Information

OpenSpine-MP: Constrained Motion Planning for Robot-Assisted Minimally Invasive Spine Surgery with Cadaveric Validation

July 9, 2026

1 Supplementary Overview

This supplementary document provides additional technical details supporting the proposed hybrid motion planning framework for robot-assisted minimally invasive spine surgery. It includes system-level specifications, detailed algorithmic formulations, extended simulation results, and additional validation analyses that complement the main manuscript.

Specifically, this document presents: (i) the system setup; (ii) an algorithmic description of the convex hull construction, geometric planning, orientation module, and sampling-based recovery components; (iii) the simulation setup and detailed results; (iv) an analysis of differences between simulation and experimental (phantom) performance; (v) the impact of ergonomic optimization on surgeon acceptance; and (vi) annotated cadaveric image.

2 System Setup

All computations were performed on a workstation equipped with an Intel Core Ultra 9 285 CPU (24 cores) and 32 GB RAM, running Ubuntu 24.04 LTS. An NVIDIA GeForce RTX 5060 GPU was available but not utilized; all computations were executed on the CPU.

The experimental setup includes the following hardware:

- **Robotic platforms:** LARA10,UR5E and UR10E(Simulaton only)
- **Tracking system:** Atracsys optical tracking camera

For convex hull computations, we used the Qhull library, which implements the Quickhull algorithm. Preoperative screw planning from CT data was performed using an in-house developed Spine Application Software (SAS).

3 Methods

3.1 Algorithm 1

This following Algorithm 1 constructs the convex hull in surgical workspace from individual pedicle screw tab geometries and conditionally incorporates the DRB hull based on proximity.

Algorithm 1 Hull Construction

Require: Number of tabs N, entry points $\{e_n\}$, docking points $\{d_n\}$, clearance l, DRB hull H_{DRB}, threshold D_{th}
Ensure: Active hull H

$R_o \leftarrow \sqrt{\frac{4l^2}{2+\sqrt{2}}}$
$\mathcal{P} \leftarrow \emptyset$
for $n = 1$ to N **do**
  $\mathbf{v}_n \leftarrow \mathbf{d}_n - \mathbf{e}_n$
  $\hat{\mathbf{v}}_n \leftarrow \mathbf{v}_n / \|\mathbf{v}_n\|$
  $\mathbf{p}_n \leftarrow \mathbf{e}_n - l \cdot \hat{\mathbf{v}}_n$
  $\Psi_n \leftarrow \textsc{FormPlane}(\hat{\mathbf{v}}_n, \mathbf{p}_n)$
  $\mathcal{V}_n \leftarrow \textsc{OctagonVertices}(\Psi_n, \mathbf{p}_n, R_o)$
  $\mathcal{O}_n \leftarrow \textsc{FormOctPrism}(\mathbf{e}_n, \mathbf{d}_n, \mathcal{V}_n, l)$
  $\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathcal{O}_n\}$
end for
$H_{\mathrm{PS}} \leftarrow \textsc{ConvexHull}(\mathcal{P})$
if $\mathrm{dist}(\mathrm{EE}, \mathrm{DRB}) < D_{\mathrm{th}}$ **then**
  $H \leftarrow \mathrm{conv}(H_{\mathrm{PS}} \cup H_{\mathrm{DRB}})$
else
  $H \leftarrow H_{\mathrm{PS}}$
end if
return H

3.2 Algorithm 2

Computes the complete path, including the inter-screw transition, by evaluating plane–hull intersections to obtain an ordered set of intersection points that define the final transition trajectory.

Algorithm 2 Path Retrieval

procedure $\textsc{RetrievePath}((d_n, e_n), (d_{n+1}, e_{n+1}), H)$
  $p\_list \leftarrow [\,]$
  for $k \in \{n, n+1\}$ **do**
    $\mathbf{v}_k \leftarrow \mathbf{d}_k - \mathbf{e}_k$
    $\hat{\mathbf{v}}_k \leftarrow \mathbf{v}_k / \|\mathbf{v}_k\|$
    $p_k^{\mathrm{PST}} \leftarrow \mathbf{e}_k - l \cdot \hat{\mathbf{v}}_k$
    if $\textsc{InsideHull}(e_k, H)$ **then**
      $\mathbf{h}_k \leftarrow \textsc{ProjectToHull}(e_k, H)$
    else
      $\mathbf{h}_k \leftarrow p_k^{\mathrm{PST}}$
    end if
    p_list.append($\mathbf{h}_k$)
  end for
  $(\mathbf{h}_n, \mathbf{h}_{n+1}) \leftarrow p\_list$
  $lineS \leftarrow$ FormLineSegment($\mathbf{h}_n, \mathbf{h}_{n+1}$)
  if $\textsc{LineIntersectsHull}(lineS, H)$ **then**
    $(\bar{\mathbf{h}}_n, \bar{\mathbf{h}}_{n+1}) \leftarrow$ LineHullIntersection($lineS, H$)
    $\mathbf{n} \leftarrow$ GraphSearchOnHull($\bar{\mathbf{h}}_n, \bar{\mathbf{h}}_{n+1}, H$)
    $\Pi \leftarrow$ FormPlane($\bar{\mathbf{h}}_n, \bar{\mathbf{h}}_{n+1}, \mathbf{n}$)
    $C \leftarrow$ PlaneHullIntersection(Π, H)
    $path_{ist} \leftarrow$ FindPath($C, \bar{\mathbf{h}}_n, \bar{\mathbf{h}}_{n+1}$)
  else
    $path_{ist} \leftarrow lineS$
  end if
  $(path_{rt}, path_{ist}, path_{ap}) \leftarrow$ GenerateCompletePath($path_{ist}, \mathbf{h}_n, \mathbf{h}_{n+1}, \mathbf{d}_n, \mathbf{d}_{n+1}$)
  $path_n^{n+1} \leftarrow (path_{rt}, path_{ist}, path_{ap})$
  return $path_n^{n+1}$
end procedure

Note: These phases define $path_{rt}$ (axial retraction), $path_{ist}$ (inter-screw transition), and $path_{ap}$ (axial approach).

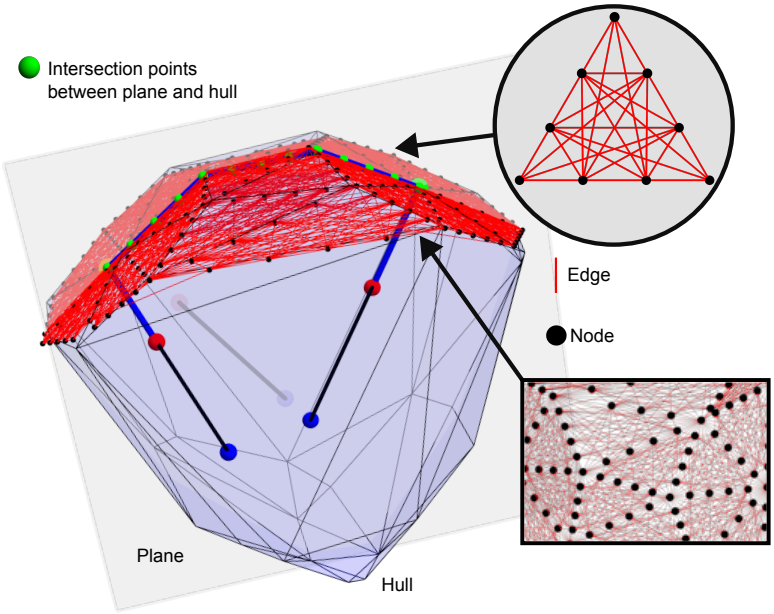

Figure 1: Convex hull–based graph construction. (**a**) Convex hull representation. (**b**) Selective graph construction over preferred hull region. (**c**) Edge subdivision and node connectivity used for discrete path planning.

3.2.1 Graph Construction on the Convex Hull

The inter-screw path on the convex hull surface H is formulated as a plane selection problem, where candidate paths are defined as intersection curves

$$\gamma(\mathbf{n}) = H \cap \Pi(\mathbf{n}),$$

with $\Pi(\mathbf{n})$ denoting a plane parameterized by its normal vector $\mathbf{n}$. The optimal path corresponds to the plane that yields the shortest surface trajectory between the transition endpoints.

To enable efficient computation, the surface H is discretized into a graph structure defined over selected regions of the convex hull (typically the upper hull). The triangulated surface mesh is refined by subdividing each triangle edge into m segments, thereby generating additional nodes along edges and within triangle interiors. These nodes are interconnected to form a dense graph that approximates the underlying surface geometry.

A heuristic-guided search is then employed to compute a discrete geodesic path between the endpoints on this graph. The resulting sequence of points provides an approximation of the shortest path constrained to the surface. From this discrete path, the optimal plane normal $\mathbf{n}$ is estimated by performing principal component analysis (PCA) on the path points, where $\mathbf{n}$ is taken as the eigenvector corresponding to the smallest eigenvalue of the covariance matrix.

This formulation effectively couples a discrete geodesic approximation with a continuous plane-parameterized representation, reducing the search space to a low-dimensional optimization problem while avoiding the computational overhead associated with exact geodesic solvers.

Table 1 summarizes the structural properties of the constructed graph. Each triangle subdivision introduces $\mathcal{O}(m)$ nodes and $\mathcal{O}(m^2)$ edges, leading to an overall graph complexity of $\mathcal{O}(Tm)$ nodes and $\mathcal{O}(Tm^2)$ edges for a mesh with T triangles. These structural properties directly govern the computational complexity of the heuristic search, influencing both the branching factor (node degree) and the size of the explored state space.

Table 1: Structural characteristics of the discretized convex hull graph as a function of the subdivision parameter m per triangle.

Graph ($m \geq 2$)	**Single Triangle**	T **Triangles**
Number of nodes	$3m$	$\leq 3Tm$
Number of edges	$3m^2 + 3m - 3$	$\leq T(3m^2 + 3m - 3)$
Minimum node degree	$2m + 1$	$2m + 1$
Maximum node degree	$3m - 1$	$\leq 3m - 1$

3.3 Algorithm 3

The Orientation Module computes kinematically feasible, collision-free TCP orientations of the end-effector along the Cartesian path at five key waypoints: three during axial approach (h_{n+1}, p_{n+1}^{PST}, d_{n+1}) and two during axial retraction (p_n^{PST}, h_n). This is formulated and solved as a constrained optimization problem, as detailed in the algorithm.

Algorithm 3 Orientation Optimization at Key Waypoints

Require: Cartesian path $\mathcal{P}$ with waypoints $\{h_n, p_n^{\mathrm{PST}}, d_{n+1}, p_{n+1}^{\mathrm{PST}}, h_{n+1}\}$, robot model, joint limits $\{q_{i,\min}, q_{i,\max}\}$
Ensure: Feasible oriented poses $\mathcal{X} = \{(x_k, R_k)\}$

$\mathcal{X} \leftarrow \emptyset$
for each waypoint $x_k \in \mathcal{P}$ **do**
  if $x_k \in \{p_n^{\mathrm{PST}}, p_{n+1}^{\mathrm{PST}}, d_{n+1}\}$ **then**
    (Case A: PST-constrained)
    Define guidance axis $\hat{g}_k$ from PST
    Parameterize orientation by θ_{PST}
    $\Theta \leftarrow \{\theta_{\mathrm{PST}}\}$
  else
    (Case B: Hull point)
    Allow full rotation space
    $\Theta \leftarrow \{\boldsymbol{\Theta}_{\mathrm{hull}} = (\theta_x, \theta_y, \theta_z)\}$
  end if
  $q^* \leftarrow \arg \min_{q \in \mathrm{IK}(x_k, \Theta)} C_{\mathrm{total}}(q)$
  $R_k \leftarrow \mathrm{OrientationfromForwardKinematics}(q^*)$
  $\mathcal{X} \leftarrow \mathcal{X} \cup \{(x_k, R_k)\}$
end for
return $\mathcal{X}$

Cost Function:
$C_{\mathrm{total}}(q) = w_e C_e(q) + w_j C_j(q) + w_s C_s(q)$
Optimization Domains:
PST-constrained: $\theta_{\mathrm{PST}} \in \mathbb{S}^1$
Hull points: $\boldsymbol{\Theta}_{\mathrm{hull}} \in SO(3)$

3.4 Algorithm 4

Infeasible trajectory segments are recovered via the sampling-based recovery as shown in Algorithm 4, decomposed into two motion regimes: (i) hull-constrained motion, where configurations are restricted to the convex hull surface with explicit collision constraints, and (ii) axial motion, characterized by constrained translational movement along the tool axis. The planner incrementally expands trees from the start and goal states and attempts connection through valid samples, enabling efficient recovery from discontinuities in the feasible set. This structured decomposition reduces the dimensionality of the search in each regime and improves convergence in narrow or cluttered regions.

Algorithm 4 Sampling-Based Recovery

Require: Infeasible segment $I \subset \text{GeoTraj}(t)$, boundary states $I_{\text{start}}, I_{\text{goal}}$, robot model
Ensure: Feasible recovery trajectory RecTraj

Extract start and goal poses:

$$(p_{\text{start}}, R_{\text{start}}) \leftarrow I_{\text{start}}, \quad (p_{\text{goal}}, R_{\text{goal}}) \leftarrow I_{\text{goal}}$$

Identify segment type of I
if $I \subset H$ **then**
    (Hull Surface Recovery)
    Define state space $\mathcal{S}_{\text{HCR}} = [\theta, \phi, R_x, R_y, R_z]^\top$
    Map $I_{\text{start}}, I_{\text{goal}} \to \mathcal{S}_{\text{HCR}}$ via inverse projection
    Initialize bidirectional trees $\mathcal{T}_s, \mathcal{T}_g$
    while connection not found **do**
        Sample (θ, ϕ) and project onto H
        Sample orientation (R_x, R_y, R_z)
        Form state $x \in \mathcal{S}_{\text{HCR}}$
        $q \leftarrow \text{IK}(x)$
        if q is collision-free and feasible **then**
            Extend $\mathcal{T}_s, \mathcal{T}_g$
        end if
    end while
else
    (Axial Retraction/Approach Recovery)
    Input: Initial and goal end-effector poses $I_{\text{start}}, I_{\text{goal}} \in SE(3)$
    Compute relative rotation:

$$R_\Delta \leftarrow R_{\text{start}}^\top R_{\text{goal}}$$

    Compute axis–angle vector:

$$\boldsymbol{\omega} \leftarrow \log(R_\Delta)$$

    Define orientation interpolation:

$$R_i(r) = R_{\text{start}} \cdot \exp(r\,[\boldsymbol{\omega}]_\times), \quad r \in [0, 1]$$

    Define state space:

$$\mathcal{S}_P(q, r) = \left[p_{\text{start}} + q(p_{\text{goal}} - p_{\text{start}}),\ R_i(r)\right]^\top$$

    Initialize bidirectional trees $\mathcal{T}_s, \mathcal{T}_g$
    while connection not found **do**
        Sample $q \in [0, 1]$, $r \in [0, \phi]$
        Construct state $x \in \mathcal{S}_P$
        $q_{\text{joint}} \leftarrow \text{IK}(x)$
        if q_{joint} is collision-free and feasible **then**
            Extend $\mathcal{T}_s, \mathcal{T}_g$
        end if
    end while
end if
RecTraj $\leftarrow$ Extract path connecting $\mathcal{T}_s$ and $\mathcal{T}_g$
return RecTraj

4 Simulation Setup and Detailed Results

This section provides additional details on the simulation environment, evaluation protocol, and extended performance results (shown in Table 2). The figures, Fig. 2 and Fig. 3, illustrate the simulation environment and the planning setup. The robot, pedicle screw tabs, and convex hull representation of the active constraint region are shown. The planned trajectory consists of axial retraction, inter-screw transition, and approach phases.

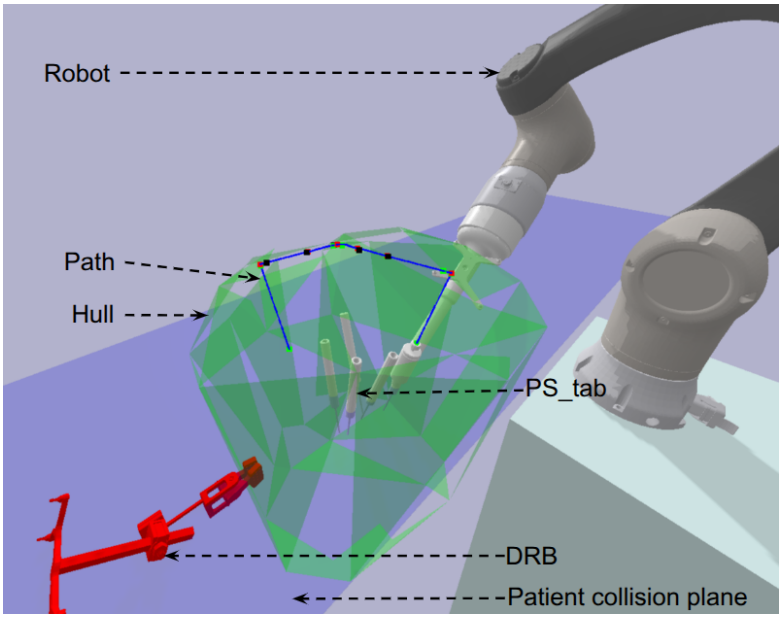

Figure 2: Visualization of constraint-aware path generation and retrieval in the simulation environment. The planned trajectory is shown in the presence of pre-existing PS_tabs, the convex hull, and the patient collision plane. The scene includes the robot, patient plane, and dynamic reference body (DRB), illustrating a feasible collision-free path generated by the planner under the modeled workspace constraints.

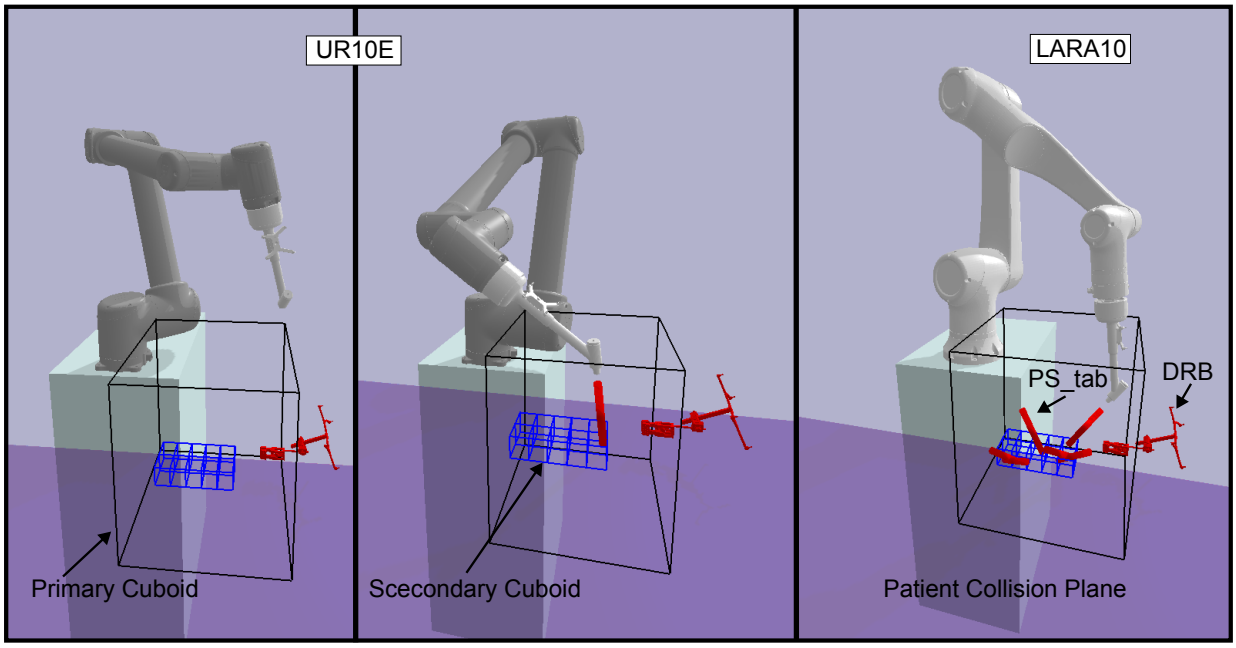

Figure 3: Simulation environment illustrating different stages of the planning process across robotic platforms. (**Left**) UR10E setup showing the initial scene prior to planning, including the primary and secondary cuboids and the dynamic reference body (DRB), defining the sampling workspace. (**Center**) Intermediate stage demonstrating pedicle screw tab (PS_tab) placement and axial retraction motion along the planned trajectory. (**Right**) LARA10 configuration after multiple PS_tab placements, showing the resulting constrained workspace and a axial retraction position.

Table 2: Summary of planning performance across planners and robots. Compute time and clearance are reported only for successful plans.

Robot	Planner	Success (%)	Compute Time (s)	Clearance (mm)
Lara10	CHOMP	76.1	1.54 ± 0.52	72.9 ± 28.3
Lara10	RRTConnect	81.2	0.11 ± 0.16	66.9 ± 30.4
Lara10	RRTstar	82.7	1.23 ± 0.96	67.3 ± 29.9
Lara10	ABITstar	86.3	0.44 ± 0.54	69.3 ± 28.5
Lara10	**Hybrid**	**92.2**	**0.07 ± 0.03**	**107.1 ± 20.7**
UR10E	CHOMP	67.8	1.65 ± 0.62	86.0 ± 36.1
UR10E	RRTConnect	72.0	0.16 ± 0.25	78.1 ± 39.1
UR10E	RRTstar	74.6	1.26 ± 0.94	78.1 ± 38.5
UR10E	ABITstar	79.5	0.51 ± 0.67	80.1 ± 36.8
UR10E	**Hybrid**	**89.9**	**0.08 ± 0.03**	**118.2 ± 28.0**

5 Simulation and Phantom Performance Comparison

The observed difference in success rates between simulation (approximately 90%) and phantom experiments (98.1%) arises from fundamental differences in evaluation scope and objective. In simulation, the planner is evaluated over a broad and largely unconstrained sampling of surgical configurations, including extreme angulations, near-singular kinematic states, and densely obstructed workspaces. This setting intentionally stresses the planner by exposing it to edge cases involving inverse kinematics infeasibility, narrow collision margins, and compounded geometric constraints. As a result, the reported success rate reflects the planner's performance over the full reachable and near-boundary configuration space. In contrast, the phantom experiments are conducted on a subset of clinically feasible plans. These scenarios are filtered through anatomical, surgical, and ergonomic constraints, thereby excluding configurations that are inherently infeasible or unsafe. Consequently, the phantom evaluation primarily measures execution reliability and collision detection fidelity within a practically relevant operating regime.

Importantly, the absence of false negatives and the correct identification of all induced collision cases in the phantom study indicate that the lower success rate in simulation is not due to deficiencies in collision modeling or execution, but rather due to the inclusion of geometrically or kinematically infeasible configurations in the simulation space.

Therefore, the higher success rate observed in phantom experiments is consistent with expectations and reflects the system's robustness under clinically admissible conditions, while the simulation results provide a more comprehensive characterization of planner behavior across the full configuration space.

6 Impact of Ergonomic Optimization on Surgeon Acceptance

Table 3 outlines the comparative surgeon acceptance rates across 160 planned scenarios with and without the integration of the ergonomic cost function. Activating the ergonomic optimization more than doubled clinical acceptance from 41% to 89% by systematically mitigating the physical site obstructions that compromised Group 1 plans. As a result, rejections in the optimized configuration (Group 2) were restricted solely to cases of extreme percutaneous hardware density, confirming the module's effectiveness in providing highly viable docking poses.

Table 3: Comparative Analysis of Surgeon Acceptance by Ergonomic Cost Function Configuration ($n = 80$ per group)

Configuration	**Accepted (n)**	**Acceptance %**	**Primary Cause of Rejection**
Group 1: Without Ergonomic cost	33 / 80	41%	Physical site obstruction
Group 2: With Ergonomic cost	71 / 80	89%	Extreme PS_tab density

7 Cadaver Supplementary Image

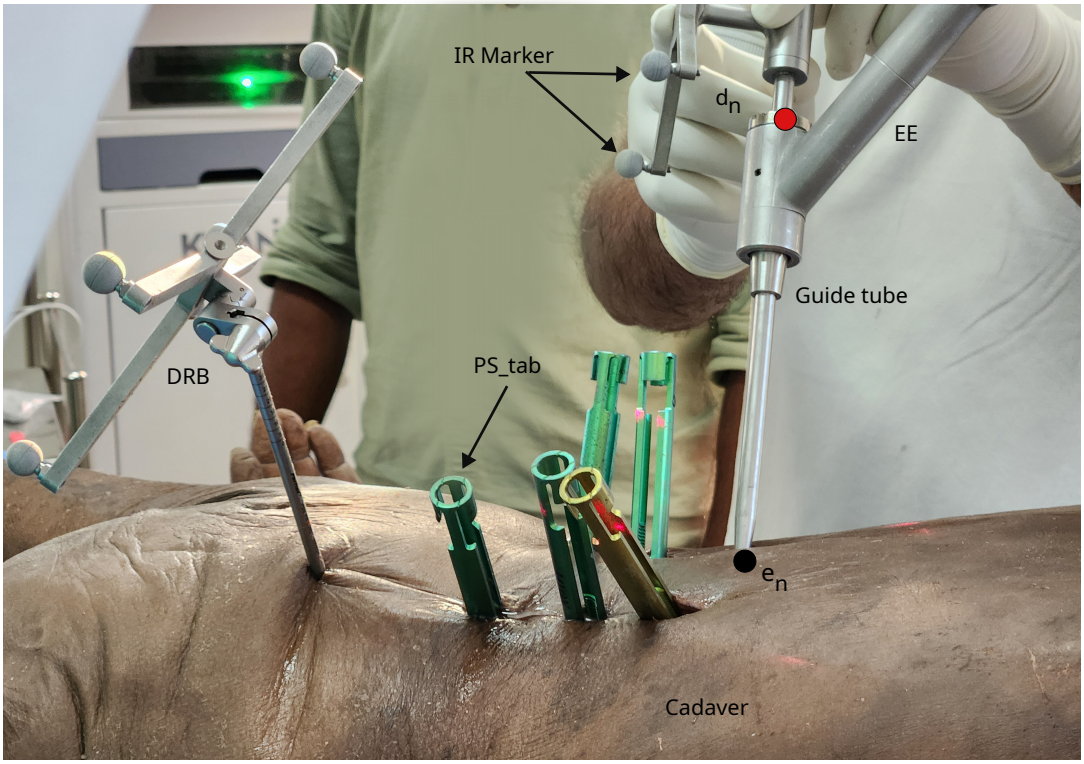

Figure 4: Cadaveric experimental setup showing the end-effector (EE) mounted on the LARA10 robot during the RA-MISS study, along with pedicle screw tabs (PS_tabs) equipped with DRB and the surgical tool.