Subject: Internship Proposal

ID	PTI_EN_distefano salvatore_08/03/2026 9.23.28
Data	08/03/2026 9.23.28

Project Supervisor

Surname	distefano
Name	salvatore
Department	MIFT
Laboratory	HPCA Lab
E-mail	sdistefano@unime.it
Phone number	

Project Co-Supervisor

Surname	
Name	
Job Position	
Department	

Laboratory	
E-mail	
Phone number	

Project details

Title	Programmable Metamaterials: Data-Driven Design, Coding, and Digital Twinning

Detailed description: 1) Rationale (why this now)

Programmable metamaterials (mechanical and electromagnetic) promise properties “on demand” (e.g., tunable stiffness, multistability, wave steering), but the field lacks comprehensive end-to-end computational tooling—a gap that CS/DS students can help close. [frontiersin.org], [nature.com]
Machine intelligence is rapidly reshaping metamaterial design—surrogate modeling, inverse design, and optimization shorten iteration cycles and open larger design spaces. [academic.oup.com]
Electromagnetic metasurfaces are already described and controlled via digital “codebooks” and space–time sequences, which maps naturally to compilers, scheduling, and firmware abstractions familiar to CS students. [cambridge.org]
Civil/structural applications (acoustic attenuation, vibration control, programmable responses) are expanding, and link well to Digital Twins for monitoring and control—another data-centric frontier with immediate tooling needs. [mdpi.com], [iris.uniroma1.it]

2) Internship learning outcomes
By the end, interns will be able to:

Model metamaterial unit cells and encode behaviors as data/parameters (geometry, states, stimuli schedules). [frontiersin.org]
Build ML surrogates and inverse-design loops to hit target effective properties or wavefront behaviors. [academic.oup.com]
Author a “coding layer” for EM metasurfaces (think: a small DSL + compiler from high-level intents to code matrices). [cambridge.org]

Integrate with a Digital Twin scaffold for simulation, monitoring, and what‑if analyses in a built‑environment context. [iris.uniroma1.it]

3) Tracks (choose one or pair two per student)
Track A — EM Metasurface Code Compiler (software‑first)
Goal: From a high‑level wave intent (e.g., “steer beam to θ, suppress sidelobes”), generate code matrices for a programmable metasurface, with a back‑end that emits microcontroller/FPGA‑ready patterns.
Core tasks:

Implement pattern generators (phase ramps, holographic phase masks), quantized to 1–4‑bit codebooks; validate via simplified array models. [cambridge.org]
Add search/optimization (e.g., evolutionary or gradient‑free) for sidelobe control; export patterns as device sequences (JSON/CSV + timing). [cambridge.org]
(Stretch) Space–time coding for dynamic beam scanning; interface stub for embedded targets. [cambridge.org]

Deliverable: metacode/ open‑source Python package + docs + examples.

Track B — Mechanical Metamaterial Inverse Design (ML + optimization)
Goal: Learn a surrogate model (e.g., regressor) mapping unit‑cell parameters → effective properties (Eeff,v,G,bandgaps)(E_\text{eff}, \nu, G, \text{bandgaps})(Eeff,v,G,bandgaps); implement inverse design to meet targets (e.g., auxetic v<0\nu<0v<0, specified stiffness window).
Core tasks:

Curate or generate parametric datasets (unit‑cell parameters, FEM results); train surrogates (GP/NN) with uncertainty estimates. [academic.oup.com]
Implement inverse design (Bayesian optimization / genetic algorithms) to reach target behaviors; visualize Pareto fronts. [academic.oup.com]
(Optional) Encode stimulus programs (temperature/strain) for multistability switching demonstrations in simulation. [frontiersin.org]

Deliverable: metaID/ notebooks, trained models, and a reproducible benchmark.

Track C — Metamaterial‑aware Digital Twin (DT) Sandbox (data + simulation)
Goal: A lightweight DT pipeline that ingests sensor or synthetic data from a meta‑enhanced component (e.g., an acoustic/vibration panel), estimates state, and runs “what‑if” simulations (e.g., retune pattern X → response Y).

Core tasks:

DT schema: define entities (geometry, states, code schedules), data adapters (CSV/MQTT), and a simple visualization dashboard. [iris.uniroma1.it]
Implement model‑based and ML‑based predictors; evaluate control policies (rule‑based → MPC stub) for performance KPIs. [tandfonline.com]
(Optional) Connect Track A/B assets as “plants” inside the DT loop. [iris.uniroma1.it]

Deliverable: metatwin/ minimal DT framework + demo scenarios.

4) Suggested timeline (from 10 to 12 weeks)
Week 1–2 — Onboarding & baselines

Readings + mini‑labs (see §9).
Reproduce a baseline: (A) a static beam‑steer code, (B) a small surrogate on synthetic data, or (C) DT skeleton with a dummy sensor stream. [cambridge.org], [academic.oup.com], [iris.uniroma1.it]

Week 3–6 — Core development

A: code generators + optimization; pattern library. [cambridge.org]
B: dataset growth + surrogate training + inverse design loop. [academic.oup.com]
C: DT data model + simulation stubs + dashboards. [iris.uniroma1.it]

Week 7–9 — Integration & evaluation

Define KPIs and test suites (e.g., beam pointing error; property hit rate; DT prediction NRMSE).
Add documentation and examples.

Week 10–12 — Packaging & dissemination

Open‑source release (MIT/Apache), short paper or extended abstract + recorded demo.

5) Student profile & skills

Python (NumPy/Pandas), basic ML (scikit‑learn/PyTorch)

6) Tech stack (all free/open)

Core: Python, Jupyter, scikit‑learn, PyTorch/Lightning or JAX, Optuna/DEAP (optimization), Plotly/Altair.
Simulation options: FEM via open‑source (for 2D trusses/plates) or simplified analytical array models for metasurfaces; CSV/MQTT stubs for DT data. (No proprietary solvers required; we stay at "teaching‑grade" fidelity.)

7) Evaluation rubric (per track)

Technical quality (40%): correctness, tests, metrics vs. baselines.
Research depth (25%): literature alignment, novelty within scope (e.g., space–time coding primitives; uncertainty‑aware inverse design). [cambridge.org], [academic.oup.com]
Software engineering (20%): code quality, docs, CI, examples.
Communication (15%): clear report, demo video, reproducibility.

8) Risks & ethics

Computational fidelity vs. reality: emphasize model scope (teaching models vs. high‑accuracy physics) and document assumptions. The literature itself flags missing comprehensive computational tools—transparency is essential. [nature.com]
Data governance: synthetic data preferred; if any real data are used, strip identifiers and comply with lab/department data policies.
Safety: no live RF transmission or mechanical prototyping is required for the internship scope.

9) Starter reading pack (1–2 hours each; accessible & current)

Mechanical metamaterials: programming strategies & properties (Frontiers, 2024).
https://www.frontiersin.org/journals/materials/articles/10.3389/fmats.2024.1361408/full

ML for metamaterial design: survey of methods, gaps, and trends (Oxford Open, 2024).
https://academic.oup.com/ooms/article/4/1/itae001/7604561

Programmable EM metasurfaces: coding & space–time control (Cambridge, 2023).
https://www.cambridge.org/core/journals/programmable-materials/article/programmable-m

etamaterials/97F91407E95B99C8CDE5A4238BF0E2E6

Metamaterials in civil infrastructure: programmable responses for acoustic/mechanical performance (MDPI, 2025). [mdpi.com]
https://www.mdpi.com/1996-1944/18/17/4032

Digital twins for AECO: enablers & frameworks (Energies, 2024; T&F, 2024).
https://iris.uniroma1.it/retrieve/67427ed1-3e6f-45a0-aee2-0151e93803d9/Piras_Digital%20Twin%20Framework_2024.pdf

10) Expected outputs (per cohort)

1–2 open-source repos (Track A/B/C), each with a tutorial notebook and test datasets.
A short working paper (6–8 pages) or extended abstract suitable for a student workshop (CS/AI or AEC tech).

11) Optional cross-track capstone (week 11–12)
"Design–Code–Twin": Use Track B to inverse-design a target mechanical behavior → emulate its effect in a simple structure; represent that component in Track C's DT; if time permits, mock an EM metasurface from Track A as a controllable boundary condition in the DT (e.g., tunable acoustic reflection).
https://www.mdpi.com/1996-1944/18/17/4032

Duration (month – max 12)	12
Duration (hours)	100
Open positions	8

Internship Skills

Technical requirements: Python

Other skills	