Workload Characterization of Programmable Metasurfaces

Taqua Saeed∗
Frederick University Cyprus
Nicosia, Cyprus

Sergi Abadal
Universitat Politècnica de Catalunya
Barcelona, Spain

Christos Liaskos
Foundation of Research and Technology (FORTH)
Heraklion, Greece

Andreas Pitsillides
University of Cyprus
Nicosia, Cyprus

Hamidreza Taghvaee
Universitat Politècnica de Catalunya
Barcelona, Spain

Albert Cabellos-Aparicio
Universitat Politècnica de Catalunya
Barcelona, Spain

Marios Lestas
Frederick University Cyprus
Nicosia, Cyprus

Eduard Alarcón
Universitat Politècnica de Catalunya
Barcelona, Spain

ABSTRACT
Metasurfaces are envisaged to play a key role in next-generation wireless systems due to their powerful control over electromagnetic waves. The last decade has witnessed huge advances in this regard, shifting from static to programmable metasurfaces. The HyperSurface (HSF) paradigm takes one step further by integrating a network of controllers within the device with the aim of adding intelligence, connectivity, and autonomy. However, little is known about the traffic that this network will have to support as the target electromagnetic function or boundary conditions change. In this paper, we lay down the foundations of a methodology to characterize the workload of programmable metasurfaces and then employ it to analyze the case of beam steering HSFs. We observe that traffic is bursty and highly dependent on the position of the target. These results will enable the early-stage evaluation of intra-HSF networks, as well as the estimation of the system performance and cost.

CCS CONCEPTS
• Computer systems organization → Embedded and cyber-physical systems; Sensors and actuators; • Networks → Network on chip.

KEYWORDS
Programmable Metasurface, Integrated Network, Traffic Analysis

∗Corresponding email: st009698@stud.fit.frederick.ac.cy

The final publication is available at ACM via https://doi.org/10.1145/3345312.3345470

1 INTRODUCTION
Metasurfaces have garnered increasing attention in recent years due to the powerful control of electromagnetic waves that they have been proven to achieve [17]. Generally composed of a periodic array of subwavelength metallic inclusions over a substrate (see Fig. 1a), metasurfaces allow to implement functionalities such as anomalous reflection, absorption, polarization control, or focusing [5, 7]. A myriad of designs have been appearing that exploit such potential for a wide range of frequencies in applications like electromagnetic cloaking, beam steering, or holographic displays [2, 8, 22, 23, 25].

Two of the main downturns of most metasurfaces are non-adaptivity and non-reconfigurability as, in most cases, the electromagnetic function and its scope are fixed once the unit cell is designed. In response to these drawbacks, metasurface design with tunable or switchable elements have emerged [15]. The resulting reconfigurable metasurfaces can be globally or locally tunable and, with the appropriate control means, they become programmable [14]. In fact, the potential of the combination of local tunability and digital control (via FPGAs, as shown in Fig. 1b) has been showcased in several prototypes of programmable metasurfaces for polarization control, focusing, or beam manipulation [4, 6, 8].

One step further towards general-purpose, autonomous, intelligent metasurfaces is the HyperSurface (HSF) paradigm [1, 10]. As shown in Fig. 1c and further detailed in Sec. 2, HSFs integrate a network of controllers within the structure of the metasurface. Controllers drive the reconfigurable unit cells and exchange information with neighbouring controllers so that the HSF can (i) implement a given electromagnetic functionality requested by an authorized user, and (ii) adapt to changes in the environment. The internal network of controllers is the enabler of the HSF approach and the main difference with respect to conventional programmable metasurfaces. This work focuses on this aspect.

To approach the design of the internal network of controllers properly, it is necessary to understand the workload that the internal HSF network will have to serve. In other words, we need to characterize the expected traffic between the HSF’s controllers in order to take informed design decisions regarding the HSF network. Due to the novelty of the programmable metasurface in general and of the HSF concept in particular, such an analysis has not been carried out thus far.
This paper aims to bridge this gap by providing a traffic analysis of programmable metasurfaces. Given the breadth of the problem, we first focus on a particular yet relevant functionality: anomalous reflection for beam steering. To this end, we develop a methodology to obtain the state of each unit cell as a function of several design parameters (e.g., unit cell size) and application parameters (e.g., steering angle). We apply the methodology to study the variations in the unit cells’ states as functions of changes in the required reflection angle. Assuming that every state change implies sending a message, these variations provide spatiotemporal information on the metasurface traffic. By mapping angular changes to physical movements in real-world scenarios, our methodology can be applied to real use cases such as tracking a mobile user in a 5G network.

The remainder of this paper is organized as follows. Sec. 2 describes the HSF paradigm. Sec. 3 outlines the system model, along with the programming rules for anomalous reflection. Sec. 4 lays down the evaluation methodology. Then, Sec. 5 presents the workload characterization results and Sec. 6 concludes the paper.

2 HYPERSURFACES: TOWARD AUTONOMOUS INTELLIGENT METASURFACES

Metasurfaces transform impinging EM waves into a response that fulfills some user-specified form. Common examples include steering a planar wave to any direction, while altering its power, polarization and phase in any joint (or disjoint) manner [7]. To attain this objective, metasurfaces rely on the Huygens principle, which states that any EM wavefront can be traced back to a planar current distribution that generates it [16]. Thus, the active elements within each meta-atom essentially force the surface currents created by the impinging EM wave into following a specific distribution that corresponds to the user-specified EM response. Towards this end, the metasurface needs to account for the EM interactions between meta-atoms, i.e., various inductive currents created due to the cross-meta-atom effects [3].

The HyperSurface paradigm dictates a framework for: i) Describing and understanding the physics behind metasurfaces in computer networking and software terms. ii) Enabling communication capabilities between metasurfaces and any other smart device via the IoT paradigm. iii) Exerting adaptive control of the EM behavior of planar materials in a classic sense-input/adapt-output loop. Thus, the HyperSurface architecture introduces three functionality layers [1, 9, 10, 14, 18]: the intra-networking layer, the gateway layer and the software control layer.

The intra-networking layer comprises a set of networked controllers embedded within the HyperSurface material. One controller is described as a head node responsible for controlling N active elements (1 : N), i.e., relaying actuation directives to each active element and gathering sensory information via measurements (e.g., power flow across active elements) or specific-purpose equipment (e.g., field sensors). Each controller has communication capabilities, i.e., at least relay information between itself and the gateway layer, while the most interesting capabilities result when inter-controller communication is also enabled [21].

We outline the following general options for the physical implementing of the controller intra-networking layer:

No intra-networking corresponds to the simplest case where each controller can communicate only with the gateway layer. For instance, each controller can be directly wired to an IoT gateway. The no intra-networking approach offers the benefits of simplicity—since the controllers either have no intelligence (e.g., simple shift registers) or are absent altogether— and central control over the HyperSurface. However, these benefits come at the expense of control scalability and single-point-of-failure concerns.

Wired intra-networking considers that there exists a wired medium that interconnects the controllers, as well as some controllers to the gateway. The wired means can refer to any nature, e.g., copper wiring, optical fiber and more. Wired networking enables the embedding of distributed intelligence within the HyperSurface, allowing the controllers to sense, cooperate and act locally, without the intervention of the gateway. This adds the benefit of limiting the physical footprint of the control elements within the HyperSurface, minimizing any degrading effects to its EM efficiency.

Wireless intra-networking assumes that each controller is equipped with a wireless transceiver module, using the very material of the HyperSurface as the communication channel. The addition of such transceivers provides the benefits of: i) zero wiring and minimal parasitic effects the EM efficiency of the HyperSurface, ii) easier HyperSurface assembly, iii) more densely connected controllers in general, reducing the network distance between controllers and facilitating faster intra-communication, and iv) natural data broadcast support, as opposed to the point-to-point inclination of the wired intra-networking option. We also highlight the existence of nano-transceivers that offer minimal complexity and are able to operate in battery-less (energy-harvesting) mode [20, 24].

The gateway layer constitutes of the hardware and communication protocols that stand between the HyperSurface and any external smart device (mobile phones, IoT devices, etc.). The gateway can be considered as a two-sided electronic module that is embedded on the HyperSurface. On the inner-side, it participates as a peer to the intra-network, being able to address controllers, sending to and receiving data from them. On the outer-side, it provides mainstream connectivity by translating the internal data to a form compatible with common protocols, such as WiFi, Ethernet, Bluetooth, etc. In this context, we define to variations of the gateway: i) The described, pass-through mode, where the gateway acts as translator of commands between the external world and the intra-controller network, and ii) the smart mode where the gateway can perform additional duties, such as the monitoring of the intra-network for errors and malfunctions, intra-routing assistance, e.g., via the smart multi-entrypoint diffusion of data.
within the intra-network for advanced performance, and handling computational duties to alleviate the controllers [19].

The software control layer comprises the software required for interacting with the HyperSurface while abstracting the underlying physics. Its role is to make HyperSurfaces easily integratable into applications and systems. To this end, it comprises of [13]: i) The HyperSurface Programming Interface (HPI), which exposes the EM capabilities of a HyperSurface in a library of software callbacks, ii) The User-side Interrupt Service, i.e., a user-side daemon that receives and handles interrupts, e.g., messages spontaneously generated by HyperSurfaces to denote hardware failures, and iii) The EM Compiler, a HyperSurface-external or -internal software service that translates the HPI callbacks to corresponding spatio-temporal actuation directives for the HyperSurface active elements [12].

At its most advanced implementation, the HSF paradigm can advance to a fully autonomic operation, where its controllers can directly measure and profile the EM cross-interactions between variable meta-atoms, i.e., ones that are in non-periodic layouts and different in shape, material composition, size, and even mobile. Subsequently, the Gateway can call upon an internal compilation service to obtain the EM behavior set by a user via the HPI.

3 SYSTEM MODEL

This section outlines the notation and main assumptions. Let us consider an HSF with $M \times N$ unit cells and a gateway as shown in Figure 2. Each unit cell is squared with side $d_u$ and can be configured in any of its $N_s$ states ($N_s = 4$ in Fig. 2(a), with each state represented by a color). The metasurface is fully illuminated by a plane wave coming from $(\theta_i, \phi_i)$ that is reflected towards $(\theta_r, \phi_r)$. The aim of the HSF is to adapt to changes in either direction.

3.1 Mobility model

Without loss of generality, let us consider that the HSF is used to keep track the position of users or objects moving along a vector $\vec{v}$. A possible application would be to maintain wireless connectivity in 5G with highly-directive antennas [11]. We study three types of trajectories, whose top view is illustrated in Fig. 2(b):

- **Case A**: the target moves in a straight line parallel to the surface where the target is at the same height than the surface. Motion starts from a point far away from the surface and finishes when the object is directly in front of it. This scenario represents the movement of mobile user, thus we assume a default speed of 1.4 m/s, which is the speed of walking of the average human.
- **Case B**: the target describes a projectile motion parallel to the surface starting from a point close to the surface and moving away from it, as illustrated in Fig. 3. This could represent a typical case of radar tracking. Unlike the horizontal movement described above, the projectile motion changes both the azimuth and elevation angles of the reflected signal. The initial speed is assumed to be 30 m/s.
- **Case C**: the target takes arbitrary leaps which results into abrupt changes in location as opposed to the gradual change in the aforementioned cases. This case represents a person moving in an area with multiple mobile obstacles resulting in intermittent connection with the surface. Using the same settings of Case A, we model the arbitrary movement by randomly changing the azimuth angle of the reflected signal.

In all cases, we consider that the metasurface is fixed onto a wall and normal incidence. The position of the moving objects can be expressed in spherical coordinates using the metasurface as point of reference. The reflected angles are obtained accordingly.

3.2 Metasurface Coding

To achieve anomalous reflection, the metasurface is modeled as a reflectarray with linear phase gradients $\Phi'_x$ and $\Phi'_y$ in the $x$ and $y$ directions, respectively, as shown in Fig. 2(a) [26]. In such a configuration, the metasurface would ideally present full reflectivity and a phase profile $\Phi(x, y) = \Phi'_x x + \Phi'_y y$ so that it complies with the required phase gradients. With this phase profile, the momentum conservation law for wave vectors at the HSF interface yields

$$k_{\parallel} \sin \theta_i \cos \phi_i + \Phi'_x = k_{\parallel} \sin \theta_r \cos \phi_r$$
$$k_{\parallel} \sin \theta_i \sin \phi_i + \Phi'_y = k_{\parallel} \sin \theta_r \sin \phi_r$$  \hspace{1cm} (1)
where \( k_i = \frac{2\pi}{\lambda_i} \) and \( k_r = \frac{2\pi}{\lambda_r} \) are the wave vectors of the incident and reflected medium with their respective wavelengths \( \lambda_i \) and \( \lambda_r \). Isolating \( \Phi'_{i'} \) and \( \Phi'_{r'} \) from Eq. (1), we can obtain the required phase gradients. Finally, since our assumed metasurface is able to tune the phase at a unit cell granularity, the phase \( \Phi_{ij} \) of the element at column \( i \) and row \( j \) has to be

\[
\Phi_{ij} = (\Phi'_{i'} + \Phi'_{r'})d_u.
\]  

(2)

Unit cells have a discrete number of states \( N_s \). Since we are interested in manipulating the phase in the \( 2\pi \) range, the unit cell is designed so that its \( n \)-th state yields a phase shift of \( \frac{2\pi}{n} \) [4]. Each unit cell will be assigned the state that is closer to the ideal phase profile \( \Phi_{ij} \).

3.3 Gateway and Controllers

Let us consider a simple HSF where each controller drives a single unit cell. For the purposes of traffic analysis, we will consider that the HSF is equipped with a gateway that:

- has external sensing capabilities to obtain the angle of incidence \( \{\theta_i, \phi_i\} \),
- communicates with other external devices to obtain the targeted direction \( \{\theta_r, \phi_r\} \),
- can compute the phase profile \( \Phi_{ij} \) and, thus, the state of each unit cell using the model above.

Whenever the target angle changes by more than a given predefined amount, which we define as angular step \( a \), the gateway computes the new phase profile and sends packets only to those unit cells that need to change its state.

4 EVALUATION METHODOLOGY

Figure 4 illustrates our methodology to obtain the traffic within the metasurface. MATLAB scripts are employed to simulate the movement and to obtain the corresponding unit cell state matrices. More specifically, we first evaluate the incidence and reflection angles given the positions of the HSF, the illumination source, and the moving target (mobility model). Then, we use Eq. (1) to obtain the phase gradients and then Eq. (2) to calculate the phase \( \Phi_{ij} \) of each unit cell. Finally, we take the unit cell state that yields the phase that is closer to \( \Phi_{ij} \) (metasurface coding). In all cases, we assume that the HSF has \( N \times M = 50 \times 50 \) unit cells.

The methods described above allow to obtain the matrix of unit cell states for any incidence and reflection conditions. Iterating over such calculations, we can obtain successive unit cell state matrices corresponding to a given movement with any angular granularity.

To model the gateway, we only obtain the unit cell state matrices in steps corresponding to the angular step parameter. A diff operation between adjacent unit cell state matrices describes which unit cells need to be changed and, given our assumptions, which unit cells will receive packets from the gateway.

For each type of movement, we collect traces that describe the time instant at which packets are generated as well as their intended destination. The traces are obtained and classified as a function of the following parameters:

- Number of states \( N_s \): the calculations used to assign cell states round to the state that yields the phase which closer to \( \Phi_{ij} \). Higher number of states corresponds to a finer resolution phase gradient that improves the metasurface performance, but possibly at the cost of transmitting more packets. By default, we set \( N_s = 4 \).
- Angular step \( a \): some applications may require tracking objects at very fine-grained angular resolution. In that case, the gateway would probably need to trigger state changes more often. This is modeled via the angular step, whose default value is set to \( a = 5^{\circ} \).

Finally, we parse the traces to obtain the following relevant metrics:

- Percentage of state-changing cells: The comparison between successive HSF state matrices determines the percentage of cells that must be adjusted to accommodate the change in the position of the target or the illumination source.
- Destination matrix: The traffic analysis yields a matrix containing the ratio of packets delivered to a given destination with respect to all the transmitted messages.

5 WORKLOAD CHARACTERIZATION

In this section, we characterize the traffic workload on the controller network. Specifically, we analyze the spatio-temporal intensity and
Workload Characterization of Programmable Metasurfaces

Figure 5: For an object moving according to Case A: (a) Reconfiguration requests during the tracking of the object. (b) Percentage of reconfigured cells versus the azimuth angle of the reflected signal. (c) The number of reconfiguration requests per second.

rate of updates generated by different types of movements. This is accomplished by showing how often reconfiguration commands are injected to the system. We use this to assess the rate of injection throughout the tracking process. In addition, we visualize the spatial distribution of the required configuration through heat maps that correspond to the surface.

The generation of reconfiguration requests relies on the location of the tracked object, the motion pattern and the angular step. While the HSF can only sense the change in the incident and reflection angles, the latter is an outcome of the distance and the height difference between the tracked object and the surface, and the motion pattern. This is depicted in Fig. 5(a) where a motion of Case A is tracked. The markers indicate the time instances when reconfiguration requests are sent from the gateway to the system. Fig. 5(a) shows that reconfigurations are more frequent as the object moves closer to the surface. For example, more than 88% of the reconfigurations are required within the last third of the motion. This is expected since the reflection angle changes faster when the object is closer to the surface.

Note that not all cells are reconfigured at each request. From Fig. 5(b), it is clear that the percentage of reconfigured cells is highly dependent on the preceding and the currently targeted angles. For example, when the tracked object moves from $\theta_r = 25^\circ$ to $\theta_r = 20^\circ$, 70% of the cells are reconfigured. When the change is from $\theta_r = 80^\circ$ to $\theta_r = 75^\circ$, on the other hand, only 28% of the cells are reconfigured in spite of the fact that the change in both cases was 5$^\circ$.

When a change in the state of the HSF is required, reconfiguration requests are streamed into the system, which renders the injection inherently bursty (i.e. a burst of reconfiguration commands are injected every time a proper change in the angle occurs). However, the number of reconfiguration requests and the frequency with which the requests are made determine the injection rate. In Fig. 5(c), we show how the injection rate significantly increases when the changes in the reflection angle are more frequent. Coincidentally, frequent changes in Case A occur for target angles affecting a higher percentage of unit cells, pushing the injection rate further. This is an important result because excessive injection rates can be an offset of congestion within the controller network.

Another way to evaluate traffic is the spatial distribution of cells to be updated. To visualize this we resort to heat maps, where hotter spots represent the regions of the HSF where controllers receive higher numbers of packets. Fig. 6 depicts the heat maps corresponding to the three scenarios of movements. Fig. 6(a) shows the case of gradual change in $\theta_r$, where the tracked object follows the motion of Case A. Fig. 6(c), however, shows the case of arbitrary changes in $\theta_r$, where the object makes sudden unpredictable leaps, i.e. Case C. The heat maps demonstrate that the traffic is almost evenly distributed over the surface for arbitrary changes. In addition, if compared with the heat maps in Fig. 6(b), one can observe the difference in the traffic distribution when the elevation angle of the reflected wave is fixed to 0 (i.e. the HSF is at the same height of the tracked object) and when it is variable. This information can be used in designing the routing mechanism, congestion control techniques and in placing the HSF tiles.

We then change some of the metasurface parameters, namely the angular step and the number of states and observe the effects on the performance. In the previous results, the angular step was set to 5$^\circ$. The value of $a$ can be used as an indication of the beam width, such that smaller values of $a$ infer narrower beam widths and higher tracking precision requirements. We investigate the impact of changing $a$ when an object moving according to Case
We have analyzed the beam steering case for different models of the HSF cells, namely $N_s = 8$ (middle) and $N_s = 16$ (right). $B$ is tracked. Case $B$ implies that both the azimuth and elevation angles are varied throughout the movement. A reconfiguration is requested every time either of the two angles change by $a^\circ$.

Fig. 7 depicts the percentage of state-changing cells corresponding to different values of the elevation angle of the reflected signal for different values of $a$. An angular step of $2^\circ$ achieves the smallest percentage of change, and thus traffic, over the entire range of angle variation, which suggests that the HSF may operate faster. However, small angular step requires a narrow beam which might increase the complexity of the surface fabrication. In addition, as the value of $a$ decreases the rate of updates is expected to increase. The effects of this increase on the injection rate will be investigated in the future. It is worth mentioning here that the movement in Fig. 7 is projectile movement. However, for clarity we only show the change in angles in the first half of the motion (until the object reaches the highest point and before dropping).

Fig. 8, on the other hand, shows the heat maps produced from tracking an object in the projectile movement for different values of $N_s$. The percentage of state-changing cells for different values of the elevation angle of the reflected signal is always zero phase. In rather large phase gradients, that area remains largely at the same state.

6 CONCLUSION

This paper has proposed and tested a comprehensive methodology for the characterization of the communication requirements of HSFs. We have analyzed the beam steering case for different models of moving targets and varying multiple design parameters. It has been observed that traffic is inherently bursty with an uneven spatial distribution and that the load increases significantly as objects approach the HSF and for HSFs with a larger number of unit cell states. Moreover, there exists a tradeoff between the rate of updates and the amount of messages per update that is regulated by the angular granularity of the tracking; the results suggest that finer tracking may result in higher yet less bursty load. In future works, we expect to use the obtained communication traces to perform a preliminary evaluation of intra-HSF networks, aiming to determine the reconfiguration capabilities of future HSFs.

ACKNOWLEDGMENTS

This work was supported by the European Union’s Horizon 2020 research and innovation programme-Future Emerging Topics (FE-TOPEN) under grant agreement No 736876.

REFERENCES


