Dual-Panel TFT Optoelectronic Computing Platform with Sensor Array for HDC, Matrix Computation, and General Optical Computing

Luigi Loreti opyoPc sa Luganp) Luigi.loreti@opyopc.com

Abstract

This document presents an optoelectronic computing platform based on **two TFT/LCD panels** and an **output sensor array**, intended as a reconfigurable architecture for multilevel optical encoding, signed-digit arithmetic, hyperdimensional computing (HDC), matrix-vector computation, and more general optical computing primitives [cite:87][cite:79][cite:88]. The central idea is to use the two TFT panels as programmable spatial modulators, the optical stack as a physical interaction stage, and the sensor array as a distributed readout layer that directly participates in computation rather than merely observing it [cite:87][cite:93].

The proposed framework is especially attractive for architectures in which spatial parallelism is more valuable than purely serial arithmetic throughput, and where local multilevel representations can simplify accumulation, recoding, similarity evaluation, and matrix-like operations [cite:79][cite:81][cite:91]. In this sense, the same hardware substrate can support different computational modes by reprogramming panel patterns, calibration lookup tables, and temporal scheduling on FPGA control logic [cite:79][cite:87].

Introduction

Liquid-crystal spatial light modulators (LC-SLMs) are now widely recognized as versatile devices for dynamically controlling optical fields in amplitude, phase, polarization, and related spatial degrees of freedom [cite:87][cite:93]. Their pixelated structure, reconfigurability, and compatibility with modern electro-optical integration make them natural candidates for computing architectures that exploit space as a computational resource rather than treating the optical field only as an imaging medium [cite:87].

In parallel, hyperdimensional computing has emerged as a robust and efficient computational paradigm based on high-dimensional distributed representations, with growing interest in architectures that can accelerate binding, bundling, similarity, and memory operations in hardware [cite:79][cite:81][cite:92]. Redundant signed-digit arithmetic and MSD-oriented processing offer another complementary direction, because they reduce the dependence on long carry propagation

chains and enable local recoding strategies that map well onto spatially parallel hardware [cite:88][cite:91][cite:85].

The platform described here combines these directions into a single optoelectronic framework. Two TFT panels provide a programmable dual-stage optical modulation process, while an output sensor array replaces a conventional global camera readout with a more local, scalable, and computation-oriented measurement plane [cite:87][cite:93]. This architecture is proposed not as a single-purpose optical arithmetic engine, but as a general research platform for multilevel optical-symbol processing, HDC primitives, and matrix-oriented computation [cite:79][cite:87].

System Architecture

The physical stack can be written as:

1. illumination source or structured optical input;
2. input polarizer, when required by the TFT/LCD operating mode;
3. **TFT Panel A**, applying the first spatial modulation stage;
4. **TFT Panel B**, aligned with Panel A and applying a second spatial modulation stage;
5. optional analyzer or output polarization-selection stage;
6. **sensor array**, aligned with the output plane for local measurement [cite:87][cite:93].

The electronic architecture includes:

- FPGA-based pattern generator;
- voltage-state calibration LUTs;
- multi-frame scheduler;
- row/column panel-driving interface;
- sensor-array acquisition frontend;
- thresholding, decoding, and reconstruction logic [cite:79][cite:92].

This decomposition is important because it separates optical physics from algorithmic control: the optical stack performs massively parallel local transformations, while the digital controller schedules states, manages calibration, and reconstructs symbols or accumulated quantities from sensor outputs [cite:87][cite:91].

Optical Model of the Dual-Panel Stack

Let $I_0(x,y)$ denote the incident optical intensity at pixel (x,y), and let $V_A(x,y)$ and $V_B(x,y)$ be the electrical drive levels applied to the corresponding pixels of Panel A and Panel B, respectively. A first-order model for the output intensity is

$$I_{out}(x,y) = I_0(x,y)\, T_{pol}\, T_A\,(V_A(x,y))\, T_B\,(V_B(x,y))$$

where T_{pol} models the polarization stack and T_A, T_B are the effective panel transmittance functions [cite:87]. A more general form, able to include polarization rotation and alignment effects, is

$$I_{out}(x,y) = I_0(x,y)\, \Phi\,(V_A(x,y), V_B(x,y), \theta_{in}, \theta_{out})$$

where Φ is the equivalent electro-optical transfer function of the cascaded system [cite:87][cite:93].

In practice, rather than relying only on a closed-form physical model, the controller uses a calibration map

$$(V_A, V_B) = \mathcal{L}(s)$$

that assigns each desired computational state s to a dual-panel drive tuple selected for optical separability and readout robustness [cite:87].

Three-State Optical Encoding

A key feature of the platform is the use of three robust optical classes, for example:

- **dark**,
- **gray**,
- **bright** [cite:87][cite:88].

These can be associated with a balanced signed-digit alphabet:

$$-1 \leftrightarrow \text{dark}, 0 \leftrightarrow \text{gray}, +1 \leftrightarrow \text{bright}$$

The sensor-array classifier then recovers the symbol by two thresholds:

$$\hat{s}(x,y) = \begin{cases} -1 & \text{if } I_{out}(x,y) < \tau_1 \\ 0 & \text{if } \tau_1 \le I_{out}(x,y) < \tau_2 \\ +1 & \text{if } I_{out}(x,y) \ge \tau_2 \end{cases}$$

with $\tau_1 < \tau_2$ chosen by calibration [cite:87][cite:88]. This three-level scheme is particularly attractive for redundant arithmetic and multilevel symbolic processing because it allows local recoding and reduces pressure on binary carry propagation [cite:88][cite:91].

Sensor Array Readout Model

For sensor element (i, j), the measured output can be modeled as

$$Y_{ij} = G_{ij}\, I_{out,ij} + O_{ij} + n_{ij}$$

where G_{ij} is the gain, O_{ij} is the offset, and n_{ij} is the equivalent readout noise. Calibration yields a normalized intensity estimate

$$\tilde{I}_{ij} = \frac{Y_{ij} - O_{ij}}{G_{ij}}$$

and, over the full sensor plane,

$$\tilde{\mathbf{I}} = \mathbf{G}^{-1}(\mathbf{Y} - \mathbf{O})$$

which can be implemented either in the sensor frontend or in FPGA post-processing [cite:79][cite:92]. Replacing a camera with a sensor array is a major architectural step because it transforms the readout plane into a distributed measurement fabric, better aligned with local optical-symbol processing [cite:87][cite:93].

Multi-Frame Computation

The architecture supports temporal decomposition into a small number of optical frames. If t denotes the frame index, then

$$V_A^{(t)}(x, y) = \mathcal{L}_A\left(s_A^{(t)}(x, y)\right), V_B^{(t)}(x, y) = \mathcal{L}_B\left(s_B^{(t)}(x, y)\right)$$

and the sensor array acquires a sequence

$$\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \ldots, \mathbf{Y}^{(T)}$$

that can be recombined digitally or used as successive optical recoding steps [cite:79][cite:87]. This frame-oriented strategy is useful for MSD arithmetic, iterative symbolic normalization, and staged accumulation [cite:85][cite:91].

MSD Arithmetic Formulation

A number in radix r and signed-digit form can be written as

$$X = \sum_{k=0}^{n-1} x_k r^k$$

where x_k belongs to a redundant alphabet such as $\{-1,0,+1\}$ [cite:88][cite:91]. Redundant and signed-digit representations are attractive because they allow arithmetic to proceed with local recoding rather than global carry chains, which is advantageous when mapping computation onto spatial hardware [cite:91][cite:85].

MSD Addition

Let

$$X = \sum_k x_k r^k, Y = \sum_k y_k r^k$$

with $x_k, y_k \in \{-1,0,+1\}$. Define the local pre-sum

$$w_k = x_k + y_k$$

where $w_k \in \{-2,-1,0,+1,+2\}$. A local recoding stage defines a digit s_k and a redundant carry-like quantity c_{k+1} such that

$$w_k = s_k + rc_{k+1}$$

with

$$s_k = \rho_s(w_k), c_{k+1} = \rho_c(w_k)$$

where ρ_s and ρ_c are implemented as LUT-based recoding functions [cite:88][cite:91].

A three-frame interpretation is natural:

- **Frame 1**: generate local pre-sums w_k;
- **Frame 2**: split them into digit and redundant transfer terms;
- **Frame 3**: normalize the result into the admissible alphabet [cite:85][cite:91].

Algorithm 1 — MSD_ADD

```
Input: digits x[k], y[k] in {-1,0,+1}
Output: digits z[k]

1. For each digit position k:
2.    w[k] = x[k] + y[k]
3.    (s[k], c[k+1]) = RECODE(w[k])
4. For each digit position k:
```

```
5.    z[k] = NORMALIZE(s[k] + c[k])
6. Return z[k]
```

The `RECODE` stage can be performed optically via dual-panel state combinations and three-level sensor classification, while `NORMALIZE` may be optical-digital or fully digital depending on implementation constraints [cite:87][cite:91].

MSD Multiplication

Multiplication is expressed as a sum of partial products:

$$P = XY = \sum_i \sum_j x_i y_j r^{i+j}$$

For ternary signed-digit alphabets, local products remain in a small discrete set, which makes them well suited to dual-panel optical generation [cite:88][cite:91].

Algorithm 2 — MSD_MUL

```
Input: digits x[i], y[j]
Output: product p[k]

1. Initialize PP[i,j] = 0
2. For each i,j:
3.    PP[i,j] = MUL_DIGIT(x[i], y[j])
4. Group terms by equal weight k = i + j
5. For each k:
6.    accum[k] = sum of PP[i,j] with i+j=k
7. Reduce accum[k] using recursive MSD_ADD
8. Return p[k]
```

Within the dual-panel architecture, `MUL_DIGIT` is realized by assigning one operand to Panel A and the other to Panel B, then classifying the resulting optical output through the sensor array [cite:87].

Matrix-Vector Computation

Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be encoded spatially on one panel and let $\mathbf{x} \in \mathbb{R}^n$ be encoded on the other panel or on the incident optical pattern. The ideal matrix-vector product is

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

with row outputs

$$y_i = \sum_{j=1}^{n} a_{ij} x_j$$

If the coefficients a_{ij} and input entries x_j are encoded as multilevel or ternary optical states, then each optical cell computes a local interaction term that is read by the sensor plane [cite:87][cite:93]. Row accumulation can be achieved by physical integration, sensor-row summation, or digital accumulation after per-pixel acquisition [cite:79][cite:92].

Algorithm 3 — MATVEC_OPTICAL

```
Input: matrix A[m,n], vector x[n]
Output: y[m]

1. Encode A on Panel A
2. Encode x on Panel B or on the input optical field
3. Illuminate the stack and acquire sensor map S[i,j]
4. For each row i:
5.    y[i] = ACCUM_ROW(S[i,1..n])
6. Apply thresholding / recoding / normalization if required
7. Return y[i]
```

This formulation highlights the main advantage of the architecture: matrix-like operators can be executed with strong spatial parallelism, while the digital backend focuses on controlled accumulation and decoding [cite:87][cite:93].

Hyperdimensional Computing Mapping

In HDC, a hypervector **h** of dimension D is written as

$$\mathbf{h} = [h_1, h_2, \ldots, h_D]$$

with binary, bipolar, or ternary components depending on the chosen model [cite:79][cite:81]. The dual-panel TFT architecture is appealing because each component can be mapped onto a spatially aligned optical cell, while the sensor array supports distributed local readout and accumulation [cite:79][cite:87].

Binding

For bipolar HDC, binding is an elementwise product:

$$\mathbf{z} = \mathbf{a} \odot \mathbf{b}, z_i = a_i b_i$$

which maps naturally to two-panel modulation [cite:79][cite:81].

Bundling

Bundling is a superposition operation:

$$\mathbf{u} = \sum_{k=1}^{K} \mathbf{h}^{(k)}$$

followed by a decision rule such as

$$\hat{u}_i = \mathrm{sgn}(u_i)$$

or by ternary three-threshold classification [cite:79][cite:81]. This is relevant because distributed sensor readout is naturally compatible with local accumulation and post-quantization [cite:79][cite:92].

Similarity

A common similarity score is the dot product:

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{D} a_i b_i$$

which again decomposes into local products and a final accumulation stage [cite:79][cite:81].

Algorithm 4 — HDC_INFERENCE

```
Input: query q, memory prototypes M[k]
Output: best index

1. Encode q optically
2. For each prototype M[k]:
3.    perform binding/similarity between q and M[k]
4.    acquire sensor map
5.    accumulate score[k]
6. Select k* = argmax_k score[k]
7. Return k*
```

This procedure is especially attractive when many high-dimensional comparisons must be evaluated under strong parallelism constraints [cite:79][cite:81][cite:92].

Unified Operator View

The overall platform can be abstracted as

$$\mathbf{s}_{out} = \mathcal{D}\left(\mathcal{Q}\left(\mathcal{S}\left(\mathcal{O}\left(\mathcal{E}_A(\mathbf{u}), \mathcal{E}_B(\mathbf{v})\right)\right)\right)\right)$$

where:

- $\mathcal{E}_A$ and $\mathcal{E}_B$ encode operands onto the two panels;
- $\mathcal{O}$ is the dual-panel optical interaction;
- $\mathcal{S}$ is the sensor-array measurement operator;
- $\mathcal{Q}$ is thresholding or quantization;
- $\mathcal{D}$ is the final digital decoding stage [cite:87][cite:93].

This representation makes clear that the sensor plane is not a passive observer but an explicit computational stage in the pipeline [cite:93].

Calibration Procedure

Correct operation depends on pixel-wise or region-wise calibration of panel response, optical output classes, and sensor readout statistics [cite:87][cite:93]. A practical calibration loop is:

Algorithm 5 — CALIBRATION

```
Input: training patterns
Output: drive LUTs and thresholds

1. Apply known level combinations to Panel A and Panel B
2. Measure output at each sensor element
3. Estimate sensor gain, offset, and noise statistics
4. Cluster responses into dark / gray / bright classes
5. Compute thresholds tau1, tau2 with maximum separation margin
6. Build LUT state -> (VA, VB)
7. Store calibration parameters in FPGA control logic
```

A margin-maximizing formulation for threshold selection is

$$\max_{\tau_1,\tau_2} \min\{\tau_1 - \mu_D, \mu_G - \tau_1, \tau_2 - \mu_G, \mu_B - \tau_2\}$$

where μ_D, μ_G, μ_B are the class centroids for dark, gray, and bright outputs.

Advantages and Limitations

The proposed architecture has three major strengths. First, it exploits **spatial parallelism** because computation is distributed across the 2D panel and sensor grids [cite:87][cite:93]. Second, it is **reconfigurable**, since the same substrate can implement different operators through FPGA-controlled patterns, frame sequencing, and LUT updates [cite:79][cite:92]. Third, it naturally supports **multilevel symbolic representations**, which are valuable for redundant arithmetic, ternary processing, and HDC-like distributed coding [cite:79][cite:88].

The main limitations are also clear. TFT/LCD response times are slower than those of faster optical modulators, pixel-to-pixel variability and thermal drift must be controlled, spatial registration between the two panels and the sensor array is critical, and classification margins can collapse if the three optical states are not well separated [cite:87][cite:93]. In addition, the overall system benefit depends on balancing optical parallelism against electronic control and readout overhead [cite:79][cite:92].

Conclusion

A platform built from **two TFT panels** and an **aligned sensor array** can be interpreted as a reconfigurable optoelectronic processor rather than as a conventional display stack [cite:87][cite:93]. Within this framework, MSD arithmetic, matrix-vector operators, HDC primitives, and more general optical computing tasks become different instances of the same underlying principle: encode information spatially, let the optical stack perform massively parallel local interactions, measure the result locally, and close the loop with digital control and decoding [cite:79][cite:81][cite:88].

The most important opportunity lies not in any single operation, but in the possibility of co-designing representation, optics, sensing, and FPGA control as one integrated computational system [cite:79][cite:87][cite:92].

References used in this document

- Kleyko et al., *A Survey on Hyperdimensional Computing aka Vector Symbolic Architectures, Part I: Models and Data Transformations* [cite:82][cite:86]
- *A Review on Hyperdimensional Computing* [cite:79]
- *Recent Progress and Development of Hyperdimensional Computing Systems for AI* [cite:92]
- *A review of liquid crystal spatial light modulators: devices and applications* [cite:87]
- *A review of liquid crystal spatial light modulators devices and applications* summary coverage [cite:93]

- Teaching material on generalized signed-digit representation [cite:88]
- Redundant representation notes on carry-free arithmetic and multiplication [cite:91]
- Analysis of digit stability in redundant number systems and MSD behavior [cite:85]

EOF && cp /home/user/output/dual_panel_tft_sensor_array_publication_en.md /home/user/output/dual_panel_tft_sensor_array_publication_en_with_refs.md