Late Breaking Results: TriSC: Low-Cost Design of Trigonometric Functions with Quasi Stochastic Computing

Sercan Aygun✦, Mehran Shoushtari Moghadam✦, and M. Hassan Najafi
University of Louisiana at Lafayette, Lafayette, LA, USA
{sercan.aygun, m.moghadam, najafi}@louisiana.edu (✦: The authors have equal contribution.) *

ABSTRACT

Low-cost and hardware-efficient design of trigonometric functions is challenging. Stochastic computing (SC), an emerging computing model processing random bit-streams, offers promising solutions for this problem. The existing implementations, however, often overlook the importance of the data converters necessary to generate the needed bit-streams. While recent advancements in SC bit-stream generators focus on basic arithmetic operations such as multiplication and addition, energy-efficient SC design of non-linear functions demands attention to both the computation circuit and the bit-stream generator. This work introduces **TriSC**, a novel approach for SC-based design of trigonometric functions enjoying state-of-the-art (SOTA) quasi-random bit-streams. Unlike SOTA SC designs of trigonometric functions that heavily rely on delay elements to decorrelate bit-streams, our approach avoids delay elements while improving the accuracy of the results. **TriSC** yields significant energy savings of up to 92% compared to SOTA. As two novel use cases studied for the first time in SC literature, we employ the proposed design for *2D image transformation* and *forward kinematics of a robotic arm*, two computation-intensive applications demanding low-cost trigonometric designs.

1 INTRODUCTION

Stochastic computing (SC) is a re-emerging computing paradigm with roots dating back to the 1960s. Despite a period of relative obscurity until the 2010s, significant interest in the paradigm surged in recent years, driven by high demand for low-cost and robust hardware designs for applications such as image processing and machine learning. SC offers low-cost and noise-robust hardware implementations for various arithmetic operations from multiplication and division to square root and exponentiation. Non-linear operations such as *trigonometric* functions are also fundamental in important applications, such as computer vision and robotics. These functions are computationally intensive, which makes them complex to implement with conventional binary architectures. This limitation has attracted attention to lightweight SC-based solutions [1, 4]. While prior work has made strides in designing SC-based *trigonometric* functions, the existing designs rely on sequential delay elements, which affect their hardware cost and latency. The designs further process pseudo-random bit-streams, demanding long bit-streams for acceptable accuracy. This study explores quasi-random bit-streams for higher accuracy, shorter latency, and higher energy efficiency. Our approach eliminates the need for delay elements. For the first time, we employ SC for the cost-efficient design of image transformation and robotic design applications, which need extensive *trigonometric* functions. The important contributions of this work are as follows:

❶ Developing **TriSC**, a low-cost and energy-efficient approach for SC-based design of *trigonometric* functions.
❷ Improving the accuracy of the SC-based *trigonometric* functions by exploiting quasi-random sources.
❸ Eliminating the need for additional decorrelators (delay elements) in the mid-stage of the SC design.
❹ Employing **TriSC** in two new and relatively intricate domains, *Image Transformation* and *2-joint Robotic Arm Manipulator*, for the first time in the SC literature.

2 BACKGROUND AND MOTIVATION

SC systems process data in the form of random bit-streams. Any scalar value in the $[0, 1]$ interval can be represented using a bit-stream with interleaved bits of '1's and '0's. The probability of observing a '1' in a bit-stream equals the scalar value. For example, any bit-stream with 20% of '1's represents 0.2 in a so-called 'unipolar' encoding. The bit positions are insignificant, different from the positional binary representation. This makes SC a fault-robust computation model. SC circuits are typically simple built from standard logic gates. For example, an AND gate is used as a multiplier for unipolar bit-streams. The cross-correlation of the input operands is important and directly impacts the accuracy of the computation. For example, for multiplication, the input bit-streams to the AND gate must be independent, i.e., uncorrelated.

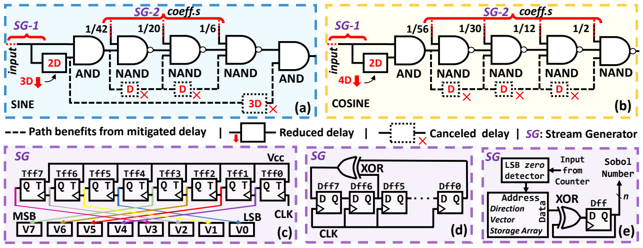

Figure 1: Proposed TriSC design: From SOTA to our proposal by eliminating the delay elements thanks to the efficient SG modules. (a) *sine* design, (b) *cosine* design, (c) VDC-2^4 example for 8-bit SG design, (d) 8-bit LFSR SG design, and (e) *n*-bit Sobol SG design.

SC systems contain three main modules. The first module generates bit-streams and typically includes one or a few random sequence generators (SGs) along with one comparator for each input. The second module is the computation circuit built from simple standard logic gates. The final module is a decoder that converts the output bit-stream back to standard binary format. For *trigonometric* functions, the second module includes a set of standard logic gates that approximate the non-linear function using the truncated Maclaurin series expansion. The state-of-the-art (SOTA) SC designs of these functions require some delay elements to decorrelate bit-streams. The proposed design of this work will minimize these delay elements. It further uses a lightweight stream generator, SG-VDC (Van der Corput Stream Generator) [3] that provides quasi-randomness for high accuracy outcomes.

3 STATE-OF-THE-ART AND PROPOSED DESIGN

Parhi and Liu [4] extensively investigated SC designs for complex non-linear functions, including *trigonometric* functions. Their design uses alternative positive and negative decreasing order coefficients for each specific polynomial function. Chu et al. [1] explored a correlation-based SC implementation of polynomial functions with unipolar bit-streams. To implement *trigonometric* functions, they propose a NAND-AND structure with fewer delay elements. In this work, we propose **TriSC**, a novel SC design for implementing *trigonometric* functions. **TriSC** applies Horner's rule to the corresponding Maclaurin series expansion. Figures 1 (a) and (b) illustrate the **TriSC**-based designs for *sine* and *cosine* functions. The key difference between our design and SOTA is in removing the mid-stage decorrelation blocks, which reduces the hardware cost and latency. **TriSC** removes these blocks by employing VDC-2^n SGs, shown in Figure 1 (c). The VDC SGs can provide the desired uncorrelation for the mid-level circuit elements. We will show that **TriSC** also archives a significantly higher accuracy compared to the SOTA designs.

The existing SC designs rely on pseudo-random sources such as Linear-Feedback Shift Registers (LFSRs) similar to Figure 1 (d) for bit-stream generation. **TriSC** utilizes quasi-random Low-Discrepancy (LD) sequences [2] for better performance. Leveraging LD sequences such as VDC and Sobol SG (Figure 1 (e)), we utilize quasi-random bit-streams in the main input (SG-1) and the scalar coefficient inputs at mid-levels (SG-2). **TriSC** generates VDC sequences using powers-of-2 bases (VDC-2^n). This enables low-cost bit-stream generation by employing a b-bit counter. Through simple *hardwiring*, the output of this counter can be tailored for any desired base, ensuring a hardware-efficient implementation (Figure 1 (c)). Utilizing VDC-2^n bases offers a unique correlation level suitable for cascaded circuit designs while also contributing to a lightweight hardware design.

Accuracy Comparison. Table 1 evaluates the accuracy of the **TriSC**-based designs compared to the SOTA. For each case, we consider all possible input values in the $[0, N]$ interval for $log_2(N)$-bit precision. We explore different SG-1,2 sources (VDC, Sobol, and LFSR) for circuit inputs and report the mean squared error (MSE). We report the total number of decorrelator delay elements at any circuit stage. As can be seen, the **TriSC**-based designs have no delay elements in the mid-level stages. The proposed design surpasses the SOTA designs, especially when the applied coefficients at SG-2 are selected from VDC-2^n bases.

[*] This work was supported in part by the National Science Foundation (NSF) under Grant 2339701 and Grant 2019511.

Table 1: Accuracy Evaluation of `TriSC` and SOTA approaches.

Func.	Design	Polyn. order	N †	Sequence type		Total # of Delay Elements				MSE
				SG-1 Input	SG-2 Coeff.s @ each mid-level stage	@ Input	@ 1^{st} stage	@ 2^{nd} stage	@ 3^{rd} stage	($\times 10^{-4}$)
sin(x)	**Prop.*** **Arch1**	7	256	VDC4 Sobol LFSR2	VDC128 VDC16 VDC256	**2**	**0**	**0**	**0**	0.576 **0.292** 0.761
	Prop.✦ **Arch2**	7	256	VDC2 Sobol LFSR2	VDC256 VDC32 VDC16	1	**0**	**0**	**0**	1.221 **0.403** 2.824
	SOTA-1* [4]	7	1024	LFSR1	LFSR3	3	1	1	3	0.361
	SOTA-2✦ [1]	7	1024	LFSR1	LFSR3	1	1	1	1	2.556
cos(x)	**Prop.*** **Arch1**	8	256	VDC8 Sobol LFSR2	VDC8,4,16,256	**2**	**0**	**0**	**0**	**1.096** 1.096 2.800
	Prop.✦ **Arch2**	8	256	VDC4 Sobol LFSR2	VDC8 VDC2 VDC256	1	**0**	**0**	**0**	1.896 1.929 **1.011**
	SOTA-1* [4]	8	1024	LFSR1	LFSR3	4	1	1	1	1.428
	SOTA-2✦ [1]	8	1024	LFSR1	LFSR3	1	1	1	1	2.942

✻: Applying the proposed method to the design of [4], ✦: Applying the proposed method to the design of [1]. †: The bit-stream lengths (N=256 for the proposed and N=1024 for SOTA) are chosen to obtain comparable accuracy. Maximal-length LFSRs are used with different polynomials. LFSR1: $x^{10}+x^{8}+x^{6}+1$, LFSR2: $x^{8}+x^{5}+x^{3}+1$, and LFSR3: $x^{10}+x^{8}+x^{5}+x^{4}$. Different VDC-2^{n} bases come from the same hardware source via simple *hardwiring*. VDC#1 shares the same base with *trigonometric* coefficients, VDC#1,#2,#3,#4 use different bases.

Table 2: Hardware Cost Comparison of `TriSC` and SOTA Designs.

Design	N	sin(x)						cos(x)					
		Area (μm^2)	CPL (ns)	Power (μW)	Energy (pJ)	ADP	EDP	Area (μm^2)	CPL (ns)	Power (μW)	Energy (pJ)	ADP	EDP
Prop. Arch1	256	**435**	0.42	**706.5**	**75.9**	**182.7**	**31.9**	**489**	0.44	**761.7**	**85.7**	**215.1**	**37.7**
Prop. Arch2	256	**430**	0.42	**666.0**	**71.6**	**180.6**	**30.1**	**479**	0.45	**705.5**	**81.2**	**215.5**	**36.5**
SOTA-1 [4]	1024	801	0.42	2178.2	936.8	336.4	393.4	883	0.42	2284.0	982.3	370.8	412.5
SOTA-2 [1]	1024	757	0.41	2042.6	857.5	310.3	351.5	852	0.42	2180.8	937.9	357.8	393.9

Prop. Arch1 is a modification of SOTA-1 [4] and Prop. Arch2 is a modification of SOTA-2 [1], respectively. The bit-stream length (N) is selected to provide comparable accuracy.

Hardware Cost Comparison. We compare the hardware cost of the proposed and SOTA designs for implementing *sine* and *cosine* functions. The SOTA designs favor LFSRs for input and mid-level bit-stream generation. However, **`TriSC`** enjoys quasi-randomness, particularly for the mid-level generators of the function coefficients. We synthesize the designs using the Synopsys Design Compiler v2018.06 with the 45nm FreePDK gate library. Table 2 reports the synthesis results in terms of footprint area, critical path latency (CPL), power consumption at maximum working frequency, energy consumption, area-delay product (ADP), and energy-delay product (EDP). As can be seen, the proposed architectures reduce area, power, and energy consumption by up to 46%, 67%, and 92%, respectively.

4 DESIGN AND IMPLEMENTATION

Next, we evaluate the performance of the proposed **`TriSC`** designs in two novel applications, which are studied for the first time in SC literature: ① *2D Image Transformation* and ② *Robot Arm Kinematics*.

2D Image Transformation is popular for QR code-based image rectification with wide usage in mobile robot navigation and camera posture calibration. These need extensive calculations involving *trigonometric* functions, demanding low-cost hardware designs [5]. Figure 2 (a) illustrates the square finder patterns of a QR code in a still image from which the reference angle is captured. QR codes are featured by three finder patterns located at the top-left, top-right, and bottom-left corners. The diagonal line (depicted in green) forms a 45° angle with the x-axis, with any deviation α from this angle indicating a rotation of the image. A transformation matrix, denoted as T, encapsulates various image manipulations, including translation, scaling, shear, reflection, and rotation. Rotation operations in T involve *trigonometric* functions such as *sine* and *cosine*. The inverse translation $\frac{1}{T}$ can correct any α rotation of an image. Conventional QR code processing also employs the inverse of T to re-transform rotated images. The goal is to revert any previously rotated image to its correct alignment during image acquisition. Figure 2 (a) illustrates the rotation-wise T matrix, which employs *sine* and *cosine* operations. The inverse operation computes the new image pixel positions, $x\_corrected$ and $y\_corrected$.

Figure 2 (b) presents the metrics we use to evaluate the performance. *Absolute Angle Error* assesses the performance of the SC designs (Proposed and SOTA) compared to the reference binary model, considering the ϵ transformation angle error. Following image correction, ϵ indicates the deviation error from the expected angle, as depicted in Figure 2 (a). *SIFT Descriptor Deviation* evaluates the deviation from scale-invariant feature transform (SIFT)-based descriptors. We compare the output images with reference images after transformation, and analyze the matched descriptors using the $\mathcal{L}1$ norm to quantify average alterations when using the implemented designs in image transformation.

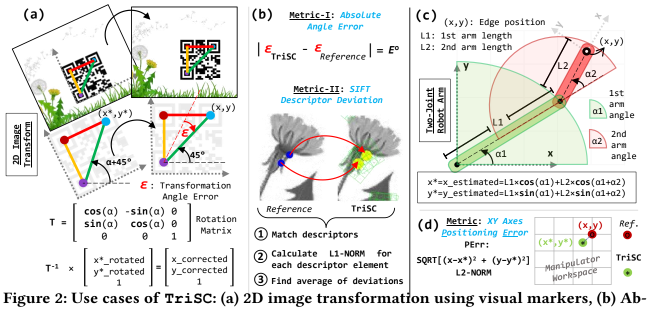
(a)
2D Image Transform
(x*,y*)
α+45°
(x,y)
ε
45°
ε : Transformation Angle Error
T = [cos(α) -sin(α) 0; sin(α) cos(α) 0; 0 0 1] Rotation Matrix
T⁻¹ × [x*_rotated; y*_rotated; 1] = [x_corrected; y_corrected; 1]
(b)
Metric-I: Absolute Angle Error
|ε_TriSC - ε_Reference| = E°
Metric-II: SIFT Descriptor Deviation
Reference
TriSC
① Match descriptors
② Calculate L1-NORM for each descriptor element
③ Find average of deviations
(c)
(x,y): Edge position
L1: 1st arm length
L2: 2nd arm length
Two-Joint Robot Arm
y
x
L1
L2
α1
α2
(x,y)
α1 1st arm angle
α2 2nd arm angle
x*=x_estimated=L1×cos(α1)+L2×cos(α1+α2)
y*=y_estimated=L1×sin(α1)+L2×sin(α1+α2)
(d)
Metric: XY Axes Positioning Error
PErr:
SQRT[(x-x*)² + (y-y*)²]
L2-NORM
(x,y)
(x*,y*)
Manipulator Workspace
Ref.
TriSC

Figure 2: Use cases of `TriSC`: (a) 2D image transformation using visual markers, (b) Absolute angle error (E°) and SIFT descriptor deviation (SIFT_{σ}), (c) Two-joint arm forward kinematics, (d) Manipulator position estimation error (*PErr*).

Table 3: Performance of `TriSC` in different use cases.

Error Calculation in Use Cases	Use-Case I: Image Orientation			Use-Case II: Robotics Positioning		
	Angle Error (E°) & SIFT Deviation (SIFT_{σ})			Position Estimation Error (PErr)		
	N=256	N=512	N=1024	N=256	N=512	N=1024
Prop. Arch1	E°: **0.091** SIFT_{σ}: **6.368**	E°: **0.084** SIFT_{σ}: **6.367**	E°: **0.068** SIFT_{σ}: **6.366**	PErr: **0.158**	PErr: **0.157**	PErr: **0.155**
Prop. Arch2	E°: **0.177** SIFT_{σ}: **6.457**	E°: **0.164** SIFT_{σ}: **6.405**	E°: **0.163** SIFT_{σ}: **6.373**	PErr: **0.176**	PErr: **0.174**	PErr: **0.173**
SOTA-1 [4]	E°: 0.386 SIFT_{σ}: 6.594	E°: 0.207 SIFT_{σ}: 6.531	E°: 0.178 SIFT_{σ}: 6.448	PErr: 0.546	PErr: 0.542	PErr: 0.541
SOTA-2 [1]	E°: 0.858 SIFT_{σ}: 6.602	E°: 0.652 SIFT_{σ}: 6.597	E°: 0.493 SIFT_{σ}: 6.543	PErr: 0.756	PErr: 0.636	PErr: 0.468

The best-performing SG-1,2 from Table 1 are used correspondingly for the Prop. Arch1 and Prop. Arch2.

We also employ the SC-based *trigonometric* designs in a *robotic kinematics* application. Figure 2 (c) illustrates the operations for a 2-joint robotic arm involving a manipulator system with two links with lengths of $L1$ and $L2$. The α_1 and α_2 angles define the movement ranges of links shown in green and red shaded regions in the xy-coordinate. The manipulator edge point (x, y) is estimated using the forward kinematics equations that involve *sine* and *cosine* functions. When SC operations are used for calculating edge positioning, we compare the estimated points (x^*, y^*) with the binary calculation as the reference to find the positioning error (*PErr*) in Figure 2 (d).

Table 3 displays the performance results for the two implemented use cases. For the first case, we examine a dataset of QR code images [6] captured in a real-world environment and report mean angle error and SIFT descriptor deviations. For the second case, we analyze navigation ranges of links for all possible Cartesian pairs of (α_1, α_2) and report the mean *PErr*. As it can be seen, across all test scenarios, our proposed architectures outperform the SOTA designs for different bit-stream lengths from 256 to 1024.

5 CONCLUSIONS

This work introduced a streamlined approach for hardware-efficient implementation of *trigonometric* functions based on SC. By leveraging quasi-random sequences, we achieve enhanced hardware efficiency owing to their straightforward design and inherent properties. Compared to SOTA, the proposed designs avoid decorrelators, resulting in lower hardware costs and latency. Our performance evaluations show higher accuracy compared to SOTA. We evaluate the performance and hardware costs for *sine* and *cosine* as two common operations, as well as two novel applications for the first time in SC literature. Our designs effectively mitigate errors in robotics vision and maneuvering, proposing a promising solution for the lightweight design of systems involving *trigonometric* functions.

REFERENCES

[1] Shao-I Chu, Chi-Long Wu, Tu N. Nguyen, and Bing-Hong Liu. 2022. Polynomial Computation Using Unipolar Stochastic Logic and Correlation Technique. *IEEE TC* 71, 6 (2022), 1358–1373.

[2] Siting Liu and Jie Han. 2018. Toward Energy-Efficient Stochastic Circuits Using Parallel Sobol Sequences. *IEEE TVLSI* 26, 7 (2018).

[3] Mehran Shoushtari Moghadam, Sercan Aygun, Mohsen Riahi Alam, Jonas I Schmidt, M. Hassan Najafi, and Nima Taherinejad. 2024. Accurate and Energy-Efficient Stochastic Computing with Van Der Corput Sequences. In *Proc. of ACM (NANOARCH '23)*. Article 27, 6 pages.

[4] Keshab K. Parhi and Yin Liu. 2019. Computing Arithmetic Functions Using Stochastic Logic by Series Expansion. *IEEE TETC* 7, 1 (2019), 44–59.

[5] Mikkel Rath Pedersen, Lazaros Nalpantidis, Rasmus Skovgaard Andersen, Casper Schou, Simon Bøgh, Volker Krüger, and Ole Madsen. 2016. Robot skills for manufacturing: From concept to industrial deployment. *Robot Comp.-Integ. Manu.* 37 (2016).

[6] István Szentandrási, Adam Herout, and Markéta Dubská. 2012. Fast detection and recognition of QR codes in high-resolution images. In *Conf. on Comp. Graph.* (Budmerice, Slovakia) *(SCCG'12)*.