Representing logic gates over Euclidean space via heaviside step function

Giovanni Iacovelli1,2,* & Claudio Iacovelli3

Theoretical concepts asserted by Alan Turing are the basis of the computation and hence of machine intelligence. Turing Machine, the fundamental computational model, has been proven to be reducible to a logic circuit and, at the same time, portable into a computer program that can be expressed through a combination of fundamental programming language control structures. This work proposes a mathematical framework that analytically models logic gates employing Heaviside Step Function. The existence of a correspondence between a generic finite-time algorithm and the proposed mathematical formulation is proven. The proposed interpretation is given through a well-defined logical circuit analytical expression. Relevant geometrical applications, related to polygon processing, having wide implications in engineering branches are presented together with a new Penalty Method for constrained optimization problems handling. A detailed simulation campaign is conducted to assess the effectiveness of the applications derived from the proposed mathematical framework.

The concept of algorithm1 was first introduced when David Hilbert posed the Entscheidungsproblem in 1928. Still, it does not have a unique definition. Essentially, it can be regarded as a finite succession of elementary instructions which leads to a result in a finite amount of time. The first and most famous mathematical interpretation was proposed by Kurt Gödel in the early '30s who introduced the notion of recursive functions2. Algorithm first characterization took place in 1936 when Alan Turing created a theoretical model, called Turing Machine (TM)3 in order to prove that Hilbert's problem was unsolvable. Under specific conditions, any function computed by TM, deterministic or not, can be reduced to a circuit which is composed by a set of Logic Gates (LGs)4. Such digital devices implement a Boolean function that takes one or more binary inputs and provides a binary output. Further, patches of circuits can be synthetized into a complicated mathematical objects. For instance, this could either be the transfer function of the electronic circuit or an object that is associated to the truth table of it. Those objects can be inputs of other gates whose definition can be extended to accept and return not only numbers but also functions. The composition of this type of gates results in the description of the whole circuit. In mathematical terms, binary status can be expressed and LGs can be realized as results of specific composition of a certain number of Heaviside Step Functions (HSFs).

This idea can also be applied to programming languages, because they provide an high-level abstraction of the machine language that is mapped into logic circuits. In fact, Böhm–Jacopini's Theorem5 states that (i) every algorithm/function can be implemented through sequence, selection and iteration (ii) every TM, up to an extent, is equivalent to a program with only compositions and iterations. This means that it is possible to associate an algorithm to a specific problem, a TM to the algorithm, a logic circuit to the TM and a function to the logic circuit. Therefore, the main goal of this article is to correlate an analytical function to a given algorithm. In fact, it will be shown how the HSF is sufficient to represent the fundamental control structures and to analytically model the LGs.

Within this framework, it is possible to reformulate analytical/geometrical problems which pertain to computer graphics, computer vision, robotics and geoscience. In these fields, one of the fundamental topics is polygons processing which is examined in several studies6–9 in literature. In10, two algorithms are designed to discover intersection points between simple polygons. The first is a set-based intersection algorithm while the second is binary-search-tree-based algorithm which allows the authors to briefly describe a set of possible merging and Boolean operations among polygons. These are also the focus of11 where the proposed algorithm is also capable to deal with polygons with holes. In12 proposes a GPU-based rasterized polygon intersection area algorithm able.
to work with complex and large-scale ones. Further, in\textsuperscript{13} an algorithm capable of performing clipping, Boolean union and difference between input polygons is presented. Based on entry/exit intersection point property, it uses an efficient data structure and can easily also be adapted to Boolean operations between regions composed by input sets. These kind of geometrical problems have also implications in medical imaging. For instance, in\textsuperscript{14} a method is developed for the analytical calculation of the 3D shape and volume of volumes' intersection in order to reconstruct 3D Positron Emission Tomography data. In Robotics geometrical information about configuration\textsuperscript{15}, placement, collision\textsuperscript{16} or proximity\textsuperscript{17} among objects are major concerns. In\textsuperscript{18}, an algorithm is proposed in order to avoid collisions between a manipulator and an obstacle. These are represented by cylinders and convex plane polygons, respectively, while the intersection among these describes the physical contact. In\textsuperscript{19}, an algorithm is designed to address another challenging aspect: motion planning. This work takes into account planar polygonal rigid robots that have to move in a polyhedral environment and avoid collisions thanks to decomposability property. The process of determining the robot's position and the obstacles places involves Point-in-Polygon Problem. In\textsuperscript{20}, instead, the authors propose an algorithm capable of solving this issue and deal with concave and convex polygons of different shapes. Throughout the article, a subset of these applications will be recast as optimization problems, via the conceived framework.

Summing up, the contributions of this work are:

- The correspondence between a finite-time algorithm and its analytical form, by means of the proposed framework, is proved. In particular, a definition of the LGs over Euclidean space is given through the HSF, namely HSFG. Further, for the sake of applicability, an equivalent smoothed form of the obtained results is also presented by employing the logistic function. An interpretation of the fundamental control structures, i.e., sequence, selection, and iteration, is also given.

- Geometrical problems previously described, together with some other geometrical applications, are tackled from an analytical perspective, thus avoiding the development of specific algorithms. Specifically, computation of volume of intersecting loci is deeply investigated to demonstrate how the mathematical construction naturally leads back to the HSFGs employment.

- A new Penalty Method to handle non-convex and even constrained optimization problems is studied. The resulting cost function requires to be probed by means of optimization techniques\textsuperscript{21} or Artificial Intelligence Search Algorithms (SAs)\textsuperscript{22}, which are able to pick the optimum or quasi-optimum solution in the output space. Intersection points finding between two implicit curves is proposed as concrete geometrical application.

Without loss of generality, Genetic Algorithm (GA)\textsuperscript{23} is used in all the investigated applications which require optimization. Nevertheless, arbitrary SA can be adopted since the proposed framework is orthogonal with respect to the algorithm used. Moreover, since the problems are handled from an analytical perspective, a direct comparison with other algorithmic solutions is not in the scope of this work. Nonetheless, conducted numerical simulations thoroughly evaluate the aforementioned contributions, in order to prove the effectiveness of the proposal. To the best of authors’ knowledge, such results have never been retrieved before.

The present work is organized as follows: “HSF Boolean logic” introduces mathematical definitions and properties of LGs realized through HSF. “Continuous HSF circuit definition” discusses continuous representation of HSF adopted. “HSF control structures” establishes a relation between algorithms and proposed gates. With “Geometrical applications”, formulation of the solutions to a set of envisioned problems is given. Section “HSF penalty method”, instead, presents a new Penalty Method for constrained optimization problems based on examined notions. Section “Numerical evaluation”, analyses numerical results derived from theoretical concepts discussed. Finally, “Conclusion” concludes the work and draws future works and research perspectives.

**HSF Boolean logic**

In this Section, a mathematical characterization is addressed to achieve a proper definition of LGs functions realized by $\Theta(x)$, i.e. HSF. Information is carried by $x$, and its binary description is achieved through a discrimination with respect to a threshold, i.e. zero value. In the most complicated case $\Theta(x)$ could not be defined for $x = 0$. On the contrary, the upcoming concepts will result in a exemplification.

**HSF circuit definition.** Without loss of generality, let the possible inputs be described with a vector space over a real scalar field, i.e. $\mathbb{R}^D$, and $O = \{x \in \mathbb{R}^D | \exists j : 1...D \ni x_j = 0\}$. Therefore, the definition of the function that encodes the real inputs in binary information is:

$$
\Theta_0 : \mathbb{R}^D \setminus O \rightarrow B^D,
$$

$$
x \rightarrow \Theta_0(x) := (\Theta(x_1), ..., \Theta(x_D))^T,
$$

where $B = \{0, 1\}$. Let $R \geq 1 \in \mathbb{N}, N_1 = D$ and $N_{R+1} = Q$. Then, it is necessary to define the following sets as:

$$
\{N_r | r : 1...R + 1, N_r \in \mathbb{N}\},
$$

$$
\{\tilde{g}_r : B^{N_r} \rightarrow \mathbb{R}^{N_{r+1}} | r : 1...R\},
$$

$$
\{\tilde{\Theta}_r : \mathbb{R}^{N_{r+1}} \setminus O_{r+1} \rightarrow B^{N_{r+1}} | r : 1...R\},
$$

where $O_{r+1}$ is the analogous of $O$ for any arbitrary input $x \in \mathbb{R}^{N_{r+1}}$, $\tilde{\Theta}_r$ acts in the same manner as $\tilde{\Theta}_0$, and $R$ is the rank, i.e. the number of compositions the HSF Circuit (HSFC) will be made of. Moreover, $\tilde{g}_r, r : 1...R$ must
be uniform continuous functions. In presence of a chain of compositions, the inner argument of $\Theta_{r+1}$ together with its kernel is:

$$\begin{align*}
\bar{h}_{r+1} &= \bar{g}_{r+1} \circ \Theta_r \circ \bar{h}_r, \ r: 1..R - 1, \ h_1 = \bar{g}_1, \\
K(h_r) &= \{ (\bar{\Theta}_0(x) \in \mathcal{B}^D) \exists i: \ldots \ldots \ 
\exists' h'_r(\bar{\Theta}_0(x)) = \emptyset \}.
\end{align*}$$

(2)

These kernels have to be excluded, because $\bar{\Theta}_0(h_r(\bar{\Theta}_0(x) \in \text{Ker}(h_r)))$ would not be defined. Obviously, the only argument that has to be checked is $h_1$ if the rank is $R = 1$. Therefore, if $K$ is the union of all kernels, the proposed definition of a HSFC $H$ of rank $R$ is:

$$H^R : \mathcal{B}^D \setminus K \rightarrow \mathcal{B}^Q,$$

(3)

$$y \rightarrow H^R(y) = \left( \bigcirc_{r=1}^R \bar{\Theta}_r \circ \bar{g}_r \right)(y),$$

(4)

where $\bigcirc_{r=1}^R(\cdot) := \bar{\Theta}_R \circ \bar{g}_R \circ \ldots \circ \bar{\Theta}_1 \circ \bar{g}_1$, and $y = \bar{\Theta}_0(x)$ is the input string. If $\Theta(0)$ is defined, then the previous definition also holds in $\mathcal{K}$. Indeed, this is a Boolean function with input $\Theta_0(x)$. The set of these functions does not have a group structure. However, the set is closed with respect to the composition operation. In the spirit of this notation, the most trivial circuit is the identity which is obtained by setting $R = 1$ and $D = Q$:

$$I : \mathcal{B}^D \rightarrow \mathcal{B}^D,$$

(5)

$$y \rightarrow \bar{\Theta}_0(y - \xi) = y,$$

derived from Eq. (4) with:

$$\bar{g}_1 : \mathcal{B}^D \rightarrow \mathbb{R}^D, \ y \rightarrow y - \xi,$$

$$\bar{\Theta}_1 : \mathbb{R}^D \setminus \mathcal{K} \rightarrow \mathcal{B}^D, \ z \rightarrow \bar{\Theta}_0(z),$$

(6)

where $\xi \in [0,1]^D$ being an arbitrary parameter which takes into account the fact that $\Theta(0)$ may not be defined or may be different from zero. This translation allows to have $\mathcal{K} = K(h_1) = K(y - \xi) = \emptyset$.

**Remark 1** Similar considerations can be done for any set of functions $\bar{f}_i : \mathbb{R}^{N_i} \rightarrow \mathbb{R}^{N_i+1} i : 0..R$, that encodes in binary the information, by substituting $\bar{\Theta}_1(\cdot)$ in the chain of composition. Indeed, $\bar{\Theta}_1(\cdot)$ can be another HSFC.

**Fundamental HSF gates.** It is worth defining an equivalent LG standard basis $\Psi_\ell = \{\text{NOT, AND, OR}\}$ with the use of Eq. (4). The HSF representation of $\Psi_\ell$ is defined as the unitary rank $(R = 1)$ basis $\Psi_H = \{N, \Lambda, \Omega\}$, constituted by the fundamental HSFGs. These particular circuits are:

- **NOT:**

$$\bar{g}_1 : \mathcal{B}^D \rightarrow \mathbb{R}^D, \ y \rightarrow \bar{g}_1(y) = (1 - y) - \xi,$$

$$\bar{\Theta}_1 : \mathbb{R}^D \rightarrow \mathcal{B}^D, \ z \rightarrow \bar{\Theta}_0(z),$$

$$\bar{\Theta}_1 \circ \bar{g}_1 : \mathcal{B}^D \rightarrow \mathcal{B}^D, \ y \rightarrow \bar{\Theta}_0((1 - y) - \xi) = I(1 - y),$$

$$\bar{\Lambda}(x) := 1 - \bar{\Theta}(x) = \bar{\Theta}(-x),$$

(7)

with $\xi \in [0,1]^D$.

- **AND:**

$$g_1 : \mathcal{B}^D \rightarrow \mathbb{R}, \ y \rightarrow g_1(y) = \prod_{i=1}^D y_i - \xi,$$

$$\Theta_1 : \mathbb{R} \rightarrow \mathcal{B}, \ z \rightarrow \Theta(z),$$

$$\Theta_1 \circ g_1 : \mathcal{B}^D \rightarrow \mathcal{B}, \ y \rightarrow \Theta\left(\prod_{i=1}^D y_i - \xi\right) = I\left(\prod_{i=1}^D y_i\right),$$

$$\Lambda(x) := \prod_{i=1}^D \Theta(x_i).$$

(8)
• OR:

\[ g_1 : B^D \rightarrow \mathbb{R}, \quad y \rightarrow g_1(y) = \sum_{i=1}^{D} y_i - \xi, \]

\[ \Theta_1 : \mathbb{R} \rightarrow B, \quad z \rightarrow \Theta(z), \]

\[ \Theta_1 \circ g_1 : B^D \rightarrow B, \quad y \rightarrow \Theta(\sum_{i=1}^{D} y_i - \xi), \]

\[ \Omega(x) := \Theta(\sum_{i=1}^{D} \Theta(x_i) - \xi), \]  \hspace{1cm} (9)

with \( \xi \in ]0, 1[ \).

As a matter of fact, for a given binary input \( y \), the output of any HSFG of \( \Psi_H \) is the same of its correspondent LG in \( \Psi_L \). Again, the absence of a definition of \( \Theta(0) \) is taken into account thanks to \( \xi \).

**Theorem 1**  Let \( y \in B^D \backslash \mathcal{K} \) be the input of a HSFG \( \mathcal{H}^B \). Then, there must exist a logic circuit \( C \) such that \( C(y) = \mathcal{H}^B(y) \) where \( C(y) \) is the output of \( C \) receiving \( y \).

**Proof**  According to the definition of \( \Psi_H \), each function of this set has an I/O relation which has an exact correspondence with the truth table of its analogous in \( \Psi_L \). Since \( \Psi_L \) can reproduce any truth table of any logic circuit \( C \) via composition of its LGs, then the same must hold for \( \Psi_H \). This means that any HSFG \( \mathcal{H}^B \) determines an I/O relation which must be reproducible through a composition of functions in \( \Psi_L \). \( \square \)

**Corollary 1**  NAND is capable of constructing any HSFC via compositions, because it must be as fundamental as \( \Psi_H \). The same also holds for NOR. Recalling Remark 1, these can be obtained by composing elements from the \( \Psi_H \) basis, in the following manner:

• NAND \( \overline{\lambda} := N \circ \Lambda \)

\[ \overline{\lambda}(x) := 1 - \prod_{i=1}^{D} \Theta(x_i). \]  \hspace{1cm} (10)

• NOR \( \overline{\Omega} := N \circ \Omega \)

\[ \overline{\Omega}(x) := \Theta(\xi - \sum_{i=1}^{D} \Theta(x_i)), \]  \hspace{1cm} (11)

with \( \xi \in ]0, 1[ \).

**Corollary 2**  DeMorgan’s Theorems are applicable to HSFGs which compose the basis \( \Psi_H \):

\[ \Lambda \circ \overline{N} = N \circ \Omega, \quad \Omega \circ \overline{N} = N \circ \Lambda. \]  \hspace{1cm} (12)

**Continuous HSF circuit definition**

The HSF is not suited for contexts and applications in which the differentiability of the functions is required. Several continuous approximations are available but, in this article, the proposed substitution is:

\[ \Theta(x) \rightarrow \sigma(\alpha x) \equiv \sigma_\alpha(x) := \frac{1}{1 + e^{-\alpha x}}, \]  \hspace{1cm} (13)

where \( \alpha \) is a coefficient that should be set sufficiently high and \( \sigma \) is the Sigmoid Function (SF), which has been widely used in engineering and physics applications\textsuperscript{24,25}. The difference between the two functions, depicted in Fig. 1, is defined as:

\[ \delta_\sigma(x) := \Theta(x) - \sigma(\alpha x) = \begin{cases} \frac{e^{-\alpha x}}{1 + e^{-\alpha x}} & \text{if } x > 0 \\ \frac{1}{1 + e^{-\alpha x}} - \frac{1}{2} & \text{if } x < 0 \\ \theta_0 - \frac{1}{2} & \text{if } x = 0 \end{cases}, \]  \hspace{1cm} (14)

where \( \theta_0 = \Theta(0) \). It can be compactly reformulated as:

\[ \delta_\sigma(x) = \frac{e^{-\alpha x}}{1 + e^{-\alpha x}} \Theta(x) - \frac{1}{1 + e^{-\alpha x}} \Theta(-x), \]  \hspace{1cm} (15)
\( \delta_0 = \delta_\alpha(0) \) is not taken into account, since in this formulation \( \delta_0 = 0 \), independently of \( \sigma_0, \delta_\alpha \) is an odd function in \( \mathbb{R}^\ast \) and it is odd in \( \mathbb{R} \) only if \( \delta_0 = 0 \): this means that (15) is odd in \( \mathbb{R} \). Moreover, \( \delta_\alpha \) goes exponentially to zero \( \forall x \neq 0 \) if \( \alpha \to \infty \). At the same time, it goes exponentially to zero \( \forall \alpha > 0 \) when \( x \to \pm \infty \) and its 1-norm is \( 2 \ln \alpha \). Therefore, the difference between \( \Theta(\cdot) \) and \( \sigma(\alpha \cdot) \) is small if \( \alpha \) is large.

The continuous version of a circuit, namely SF Circuit (SFC), is now seen as a function of a real input:

\[
\mathcal{H}_a^R: \mathbb{B}^D \rightarrow \mathbb{B}^O
\]

\[
y \mapsto \mathcal{H}_a^R(y) = (\bigcap_{t=1}^R \bar{\sigma}_{t,a} \circ \bar{g}_t)(y),
\]

where \( \mathbb{B} = [0, 1], y = (\bar{\sigma}_{0,a}(x)) \).

\[
\bar{\sigma}_{0,a}(x): \mathbb{R}^D \rightarrow \mathbb{B}^D
\]

\[
x \mapsto \bar{\sigma}_{0,a}(x) = (\sigma_a(x_1), ..., \sigma_a(x_D))^T,
\]

and \( (\bar{\sigma}_t: [0, 1]^N \rightarrow \mathbb{R}^{N_t+1}) \), is a set of uniformly continuous functions. Since the logistic function is defined \( \forall x \in \mathbb{R} \), the circuit can clearly accept any \( x \in \mathbb{R}^D \). Until the end of this Section, the following will be considered as functions of \( x \), i.e. \( \mathcal{H}^R(x) = (\mathcal{H}_a^R \circ \Theta_0)(x), \mathcal{H}_{a}^R \equiv (\mathcal{H}_a^R \circ \bar{\sigma}_{0,a})(x) \).

The uniform convergence of a SFC to its correspondent HSFCs can be proven, provided that some considerations are taken into account. The function \( \Theta_0(x) \) splits the domain in \( Q_k \) D-cubes, with \( k = 1, 2^D \) and it assumes a constant vector value within each \( Q^*_k = Q_k \setminus O \). Now let \( R \) be the rank of a given circuit \( \mathcal{H}^R \). Since any composition (which \( \mathcal{H}^R \) is made of) factorizes the domain no further than the action of \( \Theta_0 \), and since any inner argument of the composition is a constant within the reduced D-cube \( Q^*_k \), then \( \forall r \mathcal{H}_r(x) = (h_r \circ \Theta_0)(x) = \bar{c}_{r,k} \forall x \in Q^*_k \). i.e. \( \forall r | h_r|_{Q^*_k} = \bar{c}_{r,k} \). However, to have a proper definition of \( h_r \) in \( Q^*_k \), it must be that \( \forall q : 1 \leq r \leq 1: h^R_t(x) = \bar{c}^{\alpha}_t \neq 0 \forall u : 1..N_t+1, \) with \( u \) labelling the components of the vector \( h_r \). Otherwise, the action of \( \Theta_0 \) is not defined in \( Q^*_k \), implying \( \text{dom}(\mathcal{H}^R) \cap Q^*_k = \emptyset \). Therefore, if the previous happens \( \forall k : 1..2^D \), it must be concluded that \( \text{dom}(\mathcal{H}^R) = \emptyset \). This whole argument is an equivalent picture in the input domain of the kernels defined in (2).

**Theorem 2** Let \( \{\bar{\sigma}_t\}_{t} \) be a set of uniformly continuous functions, \( J = \{k : 1..2^D\} \forall r : 1..R, \forall u : 1..N_t+1 : h^{\alpha}_r|_{Q_k} = \bar{c}^{\alpha}_u \neq 0 \}, \mathcal{H}^R \) a HSFC and \( \mathcal{H}^R_a \) a SFC. Then, \( \mathcal{H}^R_{a} \) converges in

\[
\tilde{X} := \bigcup_{k \in J} X_{t,k},
\]

uniformly to \( \mathcal{H}^R \) for \( \alpha \to \infty \).

**Proof** See Appendix A.

Moreover, to get an estimation of the error committed in the evaluation of \( \mathcal{H}^R_{a}(x) \) in place of a given \( \mathcal{H}^R(x) \), the circuits may be regarded as functionals with respect to either \( \sigma_0 \) or \( \Theta_0 \). As a consequence, the notation \( \mathcal{H}(\Theta)(x) \) describes Eq. (4) while \( \mathcal{H}(\sigma_0)(x) \) describes Eq. (16). Now, the equality \( \mathcal{H}^R(\Theta) = \mathcal{H}^R(\sigma_0 + \delta_\alpha)(x) \) is trivial, and the variational Taylor series reads:
\( H^R[\Theta] = H^R[\sigma_a + \epsilon \delta_a] \bigg|_{\epsilon=1} \)

\[ = H^R[\sigma_a] + \sum_{k=1}^{\infty} \frac{\epsilon^k}{k!} H^R[\sigma_a] \bigg|_{\epsilon=0} \]

This series is convergent, and it collects both the error and the discontinuity in the functional derivatives. In the case of interest, the parameter \( \epsilon \) is set to one for the equality to hold, but the variation \( \delta_a \) has been proved to be small, so it is expected that the majority of the discrepancy has to be caused by the first orders. Finally, thanks to this analysis, it is possible to define \( \Phi_\epsilon \) basis which is equivalent to \( \Psi_H \) but exploiting the logistic function. The fundamental SF Gates (SFGs) that constitute this new basis are \( N_\alpha, \Lambda_\alpha, \Omega_\alpha \) for which Corollary 2 still holds.

**HSP control structures**

Once defined the analytical structure of a general SFG, a correlation can be established between them and algorithms.

John E. Savage proved the following, which appears as Theorem 3.8.1 of4.

**Theorem 3** Any computation performed by a one-tape Turing machine \( M \), deterministic or nondeterministic, on an input string \( w \) in \( T \) steps using \( m \) b-bit memory cells can be simulated by a circuit \( C_{M,T} \) over the standard complete basis \( \Omega \) of size and depth \( O(ST) \) and \( O(T \log S) \), respectively, where \( S = mb \) is the storage capacity in bits of \( M \)'s tape. For the deterministic TM the inputs to this circuit consist of the values of \( w \). For the nondeterministic TM the inputs consist of \( w \) and the Boolean choice input variables whose values are not set in advance.

Based on the above Theorem, the following is proved.

**Theorem 4** Let \( P \) be a program or an algorithm which computes a function in a finite amount of time. Then, there must exist an equivalent HSFG \( H \).

**Proof** Recalling Böhm–Jacopini’s Theorem, \( P \) is equivalent to TM \( M \) and can be expressed through a combination of the three fundamental control structures: sequence, selection and iteration. Therefore, reminding Theorem 3 it is possible to state that there exists a logic circuit \( C \) equivalent to \( M \). Finally, considering Theorem 1 can be claimed that there must exist a HSFC \( H \) which is equivalent to \( C \). Hence, control structures must also have a correspondent expression in these terms.

A slight interpretation is presented hereby. However, it does not pretend to be mathematically rigorous as it will considered for future investigations. The proposed characterization is proved to be sufficient for the description of control structures which compose a finite-time algorithm. For this reason, the states of all variables and operations can be expressed through Boolean strings and HSFCs, respectively. This means that a set of commands is translated into a set of HSFCs \( \{ H_{\alpha} \}_{\alpha=1}^{\infty} \), and to execute this set in sequence means:

\[ x_{OUT} = \big( \bigotimes_{\alpha=1}^{S} H_{\alpha} \big) (x_{IN}), \quad (19) \]

where \( x_{OUT} \) and \( x_{IN} \) are the state variable before and after the sequence.

The selection, instead, is inherently addressed by HSF. In fact, given logical conditions which have to be verified, this control structure can be described through a HSFC \( H \). This returns 1 if the conditions are not satisfied and 0 otherwise. The mathematic expression is:

\[ x_{OUT} = (f_1 H^R + f_0 (1 - H^R)) (x_{IN}), \quad (20) \]

where \( f_1 \) and \( f_0 \) are the statements executed in false and true cases, respectively.

Finally, for what concerns finite iteration, the definition of a recursive function turns to be necessary:

\[ x_1 = x_{IN} H^R (x_{IN}) + f (x_{IN}) (1 - H^R (x_{IN})) \]
\[ x_2 = x_1 H^R (x_1) + f (x_1) (1 - H^R (x_1)) \]
\[ \vdots \]
\[ x_n = x_{n-1} H^R (x_{n-1}) + f (x_{n-1}) (1 - H^R (x_{n-1})). \]

This synthetizes into

\[ x_n = \bigotimes_{i=1}^{n-1} f (x_{IN}) H^R \left( \bigotimes_{i=1}^{n-1} f (x_{IN}) \right) \]
\[ + \left( 1 - H^R \left( \bigotimes_{i=1}^{n-1} f (x_{IN}) \right) \right) \bigotimes_{i=1}^{n} f (x_{IN}). \]

In fact, once the \( n \)-th step is reached, all the others have to be
\[ X_k = f(X_{k-1}) = \bigcap_{i=1}^{k-1} f(x_i), \quad \forall k : 1 \ldots n - 1. \]  
(22)

If the iteration ends at the \( n + 1 \)-th step, then the output is:

\[ X_{n+1} = X_n = \bigcap_{i=1}^{n} f(x_i), \]  
(23)

and this happens if the following condition is satisfied:

\[ \forall X_{IN} \in \mathcal{L} \exists n \in \mathbb{N} \ni \mathcal{H}^k \left( \bigcap_{i=1}^{n} f(x_i) \right) = 1, \]  
(24)

with \( \mathcal{L} \subseteq \mathcal{B}^D \) being the domain of the inputs.

**Geometrical applications**

In this Section, a set of geometrical applications derived from previous theoretical discussion are deeply investigated.

**Computation of volume measure covered by intersecting geometrical loci.** The first application studied enables the computation of the volume of the union of different loci. The whole mathematical process is deeply described hereby and it is concluded with the equivalent interpretation in terms of HSFGs.

Consider a finite set of geometrical loci, described with

\[ \{ f_n(x, q_n) = 0 \mid n : 1 \ldots N \}, \]  
(25)

where \( x \) is the generic coordinate in the space \( \mathbb{R}^D \) in which the loci are embedded and \( q_n \) is a point in the vector space of parameters which specifies the locus. Addressing the point \( q_n \) with index \( n \) is equal to consider such a point and its dimensionality to be independent from all the others. It is also assumed that \( f_n \) identifies a closed and compact volume

\[ V_n = \left\{ x \mid f_n(x, q_n) \leq 0 \right\}. \]  
(26)

The aim is to find an expression which describes the volume that is effectively encompassed by the set of loci, namely

\[ V = \bigcup_{n=1}^{N} V_n = \left\{ x \mid \bigvee_{n=1}^{N} f_n(x, q_n) \leq 0 \right\}. \]  
(27)

Clearly, the major difficulty stands in the fact that the sum of the measures

\[ \mu_S = \sum_n \mu(V_n) = \sum_n \int_{V_n} dx, \]  
(28)

counts the intersections among different \( V_n \) a certain number of times, which will be later studied. Obviously, in the trivial case in which no intersections are present

\[ \mu_S = \mu(V) \iff V_i \cap V_j = \emptyset \forall i, j : 1 \ldots N, \ i \neq j. \]  
(29)

Otherwise, in order to obtain \( \mu(V) \) from \( \mu_S \), each intersecting region has to be subtracted in an appropriate way. In other words, intersections have to be counted just once. For instance, consider a \( D = 2 \) scenario in which three squares \( S_i, i : 1 \ldots 3 \) intersect in region \( R_i, i : 1 \ldots 3 \) as shown in Fig. 2. Assuming \( v \) as the measure of the unit square, \( \mu_S \) is equal to \( 48v \), while \( \mu(V) = 30v \). Defining \( \mathcal{R}_1 = S_1 \cap S_2, \mathcal{R}_2 = S_2 \cap S_3 \) and \( \mathcal{R}_3 = S_1 \cap S_2 \cap S_3 \), it results (in \( v \) units) that

\[ \mu(V) = \mu_S - \mu(\mathcal{R}_1) - \mu(\mathcal{R}_2) - 2\mu(\mathcal{R}_3) \]
\[ = 48 - 5 - 5 - 2 \times 4 = 30. \]  
(30)

As previously stated, the measure of each intersection region has to be subtracted a certain number of times that is equal to their multiplicity \( \kappa \). But first, the characteristic function of the \( n \)-th volume has to be introduced:

\[ \chi_{f_n}(x) := \Theta(-f_n(x, q_n)) \begin{cases} 
1 & \text{if } f_n(x, q_n) < 0 \\
0 & \text{if } f_n(x, q_n) > 0 \\
\vartheta_0 & \text{if } f_n(x, q_n) = 0.
\end{cases} \]  
(31)

According to the definition of the Heaviside used in this formula, the points verifying \( f_n = 0 \) count as \( \vartheta_0 \) that is usually equal to 1 or 1/2, but they also may be not counted at all if \( \vartheta_0 \) is 0 or NaN. Then, the multiplicity of a point \( x \) is defined as follows:

\[ \kappa(x) := \sum_{n} \chi_{f_n}(x) - 1. \]  
(32)

In particular, the multiplicity of a point \( x^* \) which belongs just to one volume is \( \kappa(x^*) = 0. \)
Consider a subspace \( \mathcal{M} \subseteq \mathbb{R}^D \ni V_n \subseteq \mathcal{M} \ \forall n \). Analytically, the contribution that has to be subtracted from (28) is

\[
\int_{\mathcal{M}} \kappa(x) \, dx.
\]  

(33)

However, an issue emerges in this formulation since

\[
\forall x \in \mathcal{M} \ni x \notin V : \kappa(x) = -1.
\]

In fact, each point of \( \mathcal{M} \) that does not belong to any \( V_n \) has -1 multiplicity, thus incorrectly increasing \( \mu_S \). A third addendum which rectifies this behaviour is needed:

\[
\int_{\mathcal{M}} \Theta \left( \xi - \sum_n X[f_n](x) \right) \, dx.
\]  

(34)

As a matter of fact, the integrand is equal to 1 \( \forall x \ni f_n > 0 \ \forall n \). The parameter \( \xi \) is a shift that imposes the aforementioned behaviour. Indeed, points which do not belong to any volume return a value which is different from 0 according to the adopted definition of Heaviside:

\[
\xi \in \begin{cases} 
0, 1 & \text{if } \vartheta_0 = 1 \\
0, 1/2 & \text{if } \vartheta_0 = 1/2 \\
0, 1 & \text{if } \vartheta_0 = 0
\end{cases}
\]  

(35)

Basically, as the results in (28)(33)(34) are brought together:

\[
\mu(V) = \mu_S - \int_{\mathcal{M}} \kappa(x) \, dx - \int_{\mathcal{M}} \Theta \left( \xi - \sum_n X[f_n](x) \right) \, dx.
\]  

(36)

The expression above can be rewritten in a more compact manner. It is worth noting that (33) is equivalent to

\[
\sum_n \int_{\mathcal{M}} X[f_n](x) \, dx - \int_{\mathcal{M}} \, dx = \sum_n \int_{V_n} \, dx - \int_{\mathcal{M}} \, dx.
\]  

(37)

The exchange between the sum and integral signs can be always done if \( N \) is finite, and also for \( N \rightarrow \infty \) under the hypothesis of Fubini’s Theorem. Moreover, the first integrand is equal to 1 for every point belonging to \( V_n \) at fixed \( n \) and it is 0 elsewhere. Therefore, the whole first addendum is equivalent to (28). Hence, (36) becomes:

\[
\int_{\mathcal{M}} \left( 1 - \Theta \left( \xi - \sum_n X[f_n](x) \right) \right) \, dx.
\]  

(38)

Recalling results achieved in “HSF Boolean logic”, it is clear that (38) is the following HSFGs composition:

\[
\left( N \circ \Sigma \circ \tilde{N} \right)(f_n(q_n))_n(x) = \left( \Omega \circ \tilde{N} \right)(f_n(q_n))_n(x),
\]  

(39)

which does not correspond to any element of the \( \Psi_H \) basis. However, it can be described by Algorithm 1 thanks to Theorem 4. Finally, this leads to:
\begin{align}
\mu(V) := \mathcal{V} &= \int_{\mathcal{M}} \Omega\left(\left\{ N(f_n) \right\}\right)(x)dx \\
&= \int_{\mathcal{M}} \Theta\left( \sum_n X\left[f_n\right](x) - \xi \right)dx.
\end{align}  

which perfectly describes (27). This expression, as stated in Corollary 2, is also equivalent to

\[ \mathcal{V} = \int_{\mathcal{M}} \Lambda\left[ \{f_n\} \right](x)dx. \]

**Remark 2** The characteristic function may not be defined on the locus \( f_n = 0 \) due to the definition of \( \theta_0 \). Nonetheless, the measure of a volume does not depend on the value assumed by \( \theta_0 \) because such value concerns the border which is a zero measure set.

**Remark 3** It is worth noting that is sufficient to change the sign of the argument of all HSFs in Eq. (41) to characterize the dual formula for the computation of the intersection among all volumes, that is:

\[ \mathcal{V}^* = \bigcap_{n=1}^N V_n = \left\{ x \left\vert \bigwedge_{n=1}^N f_n(x, q_n) \leq 0 \right\} \right. \]

\begin{align}
\mu(V^*) := \mathcal{V}^* &= \int_{\mathcal{M}} \Theta\left( \xi - \sum_n \Theta(f_n(x, q_n)) \right)dx \\
&= \int_{\mathcal{M}} \Lambda\left[ \{f_n\} \right](x)dx.
\end{align}

Again, recalling Corollary 2, an equivalent expression is

\[ \mathcal{V}^* = \int_{\mathcal{M}} \Lambda\left[ \{N(f_n)\} \right](x)dx, \]

that well describes (43).

**Algorithm 1** Measure of the Volume

```
Input: \{f_n\} 
Output: \mathcal{V}
1: for i = 1..M do 
2: if \( f_1(x_i) < 0 \) or \( f_2(x_i) < 0 \) ... or \( f_n(x_i) < 0 \) then 
3: \( \mathcal{V} \leftarrow \mathcal{V} + \rho^2 \) \quad \triangleright \rho \text{ stands for mesh precision} 
4: end if 
5: end for 
```

**Error analysis.** The error which is introduced by the use of the logistic function in place of Heaviside is addressed hereby. First, it is necessary to introduce the expression of the error:

\[ \delta \mathcal{V} := \int_{\mathcal{M}} \left[ \Theta\left( \sum_n \Theta(-f_n(x, q_n)) - \xi \right) \\
- \sigma\left( \sum_n \sigma(-\alpha f_n(x, q_n)) - \xi \right) \right] dx. \]

Considering (18), introduce the variable \( z = \alpha \sum_n (\sigma_n + \varepsilon \delta_n)(-f_n) - \xi \) and consider that any term \( \frac{d^k}{dx^k} z, k \geq 2 \) is zero, therefore it will not be written. As the same time, it can be reasonably assumed that the perturbation is zero at the border and, as a consequence, terms which are border-related are zero too.

Hence, it can be written:
\[
\frac{d}{d\epsilon} V = \int dx \left( \frac{d}{dz} \sigma_\alpha(z) \frac{d}{d\epsilon} z + \delta_\alpha(z) + \epsilon \frac{d}{dz} \delta(z) \frac{d}{d\epsilon} z \right),
\]
\[
\frac{d^2}{d\epsilon^2} V = \int dx \left( \frac{d^2}{dz^2} \sigma_\alpha(z) \left( \frac{d}{d\epsilon} z \right)^2 + 2 \frac{d}{dz} \delta(z) \frac{d}{d\epsilon} z \right.
+ \left. \frac{d^2}{dz^2} \delta(z) \left( \frac{d}{d\epsilon} z \right)^2 \right),
\]
\[
\frac{d^3}{d\epsilon^3} V = \int dx \left( \frac{d^3}{dz^3} \sigma_\alpha(z) \left( \frac{d}{d\epsilon} z \right)^3 + 2 \frac{d^2}{dz^2} \delta(z) \left( \frac{d}{d\epsilon} z \right)^2 \right.
+ \left. \frac{d^3}{dz^3} \delta(z) \left( \frac{d}{d\epsilon} z \right)^3 \right),
\]
\[
\frac{d^k}{d\epsilon^k} V = \int dx \left( \frac{d^k}{dz^k} \sigma_\alpha(z) \left( \frac{d}{d\epsilon} z \right)^k \right.
+ \left. 2 \frac{d^{k-1}}{dz^{k-1}} \delta(z) \left( \frac{d}{d\epsilon} z \right)^{k-1} + \frac{d^k}{dz^k} \delta(z) \left( \frac{d}{d\epsilon} z \right)^k \right).
\]

Moreover, \( \frac{d}{d\epsilon} \delta(z) = - \frac{d}{d\epsilon} \sigma_\alpha(z) \) because of the definition of \( \delta_\alpha \), so the first and third addenda for \( k \geq 2 \) simplify at every term. Let \( z' \) be equal to \( z(\epsilon = 0) = \sum \sigma_\alpha(-f_n) - \xi \) and manipulate the following:
\[
\frac{d}{d\epsilon} V + \frac{d^2}{d\epsilon^2} \frac{1}{2} V\bigg|_{\epsilon=0} = \int dx \left( \delta_\alpha(z) + 2 \frac{d}{d\epsilon} \delta(z) \frac{d}{d\epsilon} z \right). 
\]

Finally, as \( \frac{d}{d\epsilon} z = \alpha \sum \delta(-f_n) \) and the index of the Taylor series is moved back, it results that
\[
\delta V = \delta(z') - 2 \sum_{j=1}^{\infty} \alpha_{(j)}(z') (a \sum \delta(-f_n))^{j}.
\]

Custom D-dimensional volumes. Another interesting application of Heaviside is the possibility to represent customizable volumes. In fact, the technique presented in previous Section is not limited just to analytical functions. Without loss of generality, consider \( D = 2 \). Suppose that this desired area is defined by a closed polygonal chain of \( N \) segments, each of the segment lying on a given line described by \( f_n \) (or the vertical line \( x = c \), where \( c \) is a constant value). Clearly, \( \Theta(f_n) \) is a binary representation of the inequality \( f_n \geq 0 \), because the equation \( \Theta(f_n) = 1 \) is satisfied only in a subspace of \( \mathbb{R}^2 \). Capitalizing the concept of AND HSF described in "HSF Boolean logic", the customizable area is cast with the following characteristic function:
\[
\chi_\Theta = \prod_{n=1}^{N} \Theta(f_n) = \Lambda \left[ \{ f_n \} \right]. 
\]

This concept does not rely on choice of \( f_n \) as these can be anything, as long as \( \chi_\Theta = 1 \) encompasses a non-zero finite area. It is worth noting that with this formulation the Point-in-Polygon problem is inherently addressed by evaluating (50) in the desired location.

Remark 4 Reminding results of "Continuous HSF circuit definition", also for this application is possible to define an equivalent SF form \( \chi_\alpha \), taking advantage of \( \Lambda_\alpha \).

Remark 5 It should be noticed that
\[
\Omega(\{N(f_n)\}_n, \{\Lambda_\rho(\{f_m(m(q))_q\}_q) \quad \forall n, q, m(q), 
\]
represents a general partition of input domain into a certain number of disjointed sub-domains of arbitrary shape.

Custom D-dimensional borders. According to the formulation proposed so far, a further important result is achieved when the objective is not to calculate the volume but to have an expression of its border. Once again, it is sufficient to exploit the AND HSF. It is necessary that \( \delta_0 \neq 0 \) to distinguish the locus of interest in the following manner:
\[
\Gamma : \Theta(\chi_\Theta - \delta_0) \Theta(\delta_0 - \chi_\Theta) - \delta_0^2 = 0,
\]
where \( \chi_\Theta \) is the generic characteristic function which can be composed in one of the several ways previously discussed. In practice, when two face-to-face Heavisides are multiplied, the only geometrical locus that has non-zero value is the desired border. In other words, an equality constraint is imposed as will be seen in the next Section. In particular, if \( \delta_0 \neq \{0, 1\} \) then it is just needed:
However, it must be noticed that if $Y/\Theta_1$ is composed of more than one function, and their borders intersect, then Eqs. (52) and (53) are not satisfied at the intersections of borders, except for Eq. (52) when $\vartheta_0 = 1$.

Another way to realize this locus is the substitution defined in (13) so that previous formulation is equivalent to:

$$\text{where} \ Y_\sigma \ \text{is the characteristic function expressed via SF. Still, intersections do not belong to} \ \hat{W}. \ \text{If the limit is not taken, an interesting behaviour arises in the proximity of intersections due to the analyticity of} \ \sigma:\n$$

For instance, consider the configuration in Fig. 3. It consists of a square centered in $[2 2]^T$ and a circle in $[1 0]^T$ which intersect in $[1 1]^T$. Therefore, the $Y_\sigma$ in Eq. (55) is the sum of the characteristic functions of the square and the circle. The former is made with Eq. (50) and the usual substitution (13), while the latter is done with Eq. (31).

As $\alpha$ increases, $\hat{W}_\alpha$ gets closer to the real intersection. The presence of the error owed to $\alpha$ results to be negligible, and even advantageous: the proposed formulation represents an optimum analytical approximation of the union border, which in principle (i.e. the exact border) should have no defined gradient at the intersection points.

**HSF penalty method**

In this Section, an alternative representation of constrained-optimization problems is proposed. This is possible thanks to binary logic granted by HSF. Let us have a cost function $A(q)$ that has to be minimized and it is subject to some equality and inequality constraints

$$\min A(q)$$

$$h_j(q) = 0, \ \forall j \in (1..\ell)$$

$$g_i(q) \leq 0, \ \forall i \in (1..m).$$

Let’s pretend, for a while, that the minima of $A$ are different from zero. This hypothesis does not represent a restriction and the lack of it will be handled hereafter. Indeed, usually the cost function has some practical and physical meaning, therefore those minima reasonably lie within a known range. Assuming also that $\vartheta_0 = 1$, it is possible to write the following expression:

$$F(q) = A(q) \prod_{i=1}^{\ell} \Theta(-g_i(q)) \prod_{j=1}^{\ell} \Theta(h_j(q)) \Theta(-h_j(q)).$$

As in Penalty Methods, the problem is reformulated as an unconstrained one by taking into account a so-called penalty function. The proposed approach takes care of constraints by exploiting AND HSG. Indeed, previous formulation is equivalent to:

$$F(q) = A(q) \Lambda \{N(g_i)\} \Lambda \{h_j\} \Lambda \{N(h_j)\}$$

$$= A(q) \Pi \{g_i\} \Lambda \{h_j\} \Pi \{h_j\}. \quad (57)$$
As a matter of fact, the resultant function is zero across the whole space where they are not fulfilled, and it coincides with the cost function itself when they are satisfied. Moreover, each equality constraint is enabled through an AND HSFG having in input two face-to-face inequalities. The requirement \(\delta_0 = 1\) which was stated before is herein clarified. With this assumption, gates return 1 when the weak inequalities \(g_i(q) \leq 0\) are satisfied. As a consequence, it also allows the equality constraints \(h_i(q)\) to actually work. Otherwise, in both cases zero is provided. This behaviour reproduces the binary logic where the two levels are 0 and \(A(q)\) itself. Thus, the usage of the first level has to be denied. Indeed, if some minima are expected to be zeroes of \(A\), it is just needed to shift \(\mathcal{F}\) sending \(A\) into \(A + c\). Clearly, the minimal obtained values have to be shifted back after they have been found.

The composition \(\mathcal{F}\) cannot be globally convex because any of its second-order partial derivatives involves the second derivative of SF with respect to its argument. Therefore, convex optimization methods are not as well-suited as Heuristic SAs. However, these could need to move in a smooth manner between the two spaces where the constraints are satisfied or not. To this aim, once again, substitution (13) is necessary and hence SFGs have to be used instead of corresponding HSF forms. In this manner, an exponential slope is given to \(\mathcal{F}\) in which the constraints are satisfied or not. To this aim, once again, substitution (13) is necessary and hence SFGs have to be used instead of corresponding HSF forms. In this manner, an exponential slope is given to \(\mathcal{F}\), and it will depend on \(\alpha\). In particular, each equality constraint has to be multiplied by a factor 4 because the two \(\sigma\) of which is composed of would give 1/2 each, when it is satisfied. The obtained function \(\mathcal{F}\) and the optimization process are described by the pseudo-code in Algorithm 2.

### Algorithm 2 Optimization Process

| Input: \(A, \mathbb{Q}, g, h\) |
| Output: \(q^\star\) |
| Function: \(\mathcal{F}(A, q \in \mathbb{Q}, g, h)\) |
| 1: \(\text{SA}(\mathcal{F}, \mathbb{Q})\) |
| 2: |
| 3: function \(\mathcal{F}(A, q \in \mathbb{Q}, g, h)\) |
| 4: if \(g_i(q) < 0 \ldots \) and \(g_m(q) < 0\) and \(h_j(q) = 0 \ldots\) and \(h_r(q) = 0\) then |
| 5: return \(A(q)\) |
| 6: end if |
| 7: return 0 |

\(\text{or asymptotically equal to } A(q) e^{-\alpha}\) |

This method represents an alternative with respect to standard optimization, because the constraints are included in such a way that they are not constraints anymore, but a part of the whole function. Therefore, it is possible to avoid the check of the Karush–Kuhn–Tucker\(^{27}\) necessary conditions, because they rely on the construction of the Lagrangian while \(\mathcal{F}\) is formally unconstrained. Thus, computation of weights concerning the classical inclusion of constraints is avoided. Most importantly, as a final remark, this method allows constraints also to be put in any other logical relation.

### Curves intersections

A useful geometrical application of proposed HSF Penalty Method (HSFPM) is to find how many and which intersections there are when two or more curves lie in the same locus. The methodology presented hereby is suitable in the case of two curves, but can be iterated in case of more. Consider \(f_1, f_2\) and \(X = \{x_i \in \mathbb{R} : i = 1 \ldots N_{\text{MAX}}\}\) and \(Y = \{y_i \in \mathbb{R} : i = 1 \ldots N_{\text{MAX}}\}\), where \(N_{\text{MAX}}\) is the maximum intersections number (see Bézout’s Theorem\(^{28}\)). The optimization problem can be stated as follows

\[
(P1) \min_{X,Y} \sum_{i=1}^{N_{\text{MAX}}} f_1(x_i, y_i)^2 + f_2(x_i, y_i)^2 \text{ s.t.}
\]

\[
\left\| [x_k y_k]^T - [x_j y_j]^T \right\| \leq \zeta \quad \forall k \in (1, N_{\text{MAX}} - 1)
\]

\[
\forall j \in (k + 1, N_{\text{MAX}}),
\]

where \(\zeta\) is a user-defined minimum distance to distinguish intersection points. This is a non-convex optimization problem that is hard to solve even with mathematical approximation. The technique proposed in this Section can be used to reformulate the problem as:

\[
(P2) \max_{X,Y} \sum_{i=1}^{N_{\text{MAX}}} \Theta (-f_i^*) \prod_{k < j} \mathcal{J}_{kj},
\]

where

\[
\mathcal{J}_{kj} = \Theta \left( \left\| [x_k y_k]^T - [x_j y_j]^T \right\| - \zeta \right),
\]

\[
f_i^* = f_1^2(x_i, y_i) + f_2^2(x_i, y_i).
\]

A SA can be used to solve this problem.

Basically, every intersection point corresponds to a maxima of the objective function. Moreover, a \(\Theta(\cdot)\) has been introduced in the objective function derived from \((P1)\) in order to assure that local maxima are avoided in \((P2)\), and the function value will now be equal to the number of intersections which the algorithm finds. In
The worst case scenario is when the algorithm tries to count a point at a distance $\zeta$ from a true intersection as if it was a real one. The requirement in Eq. (58) prevents such unwanted behaviour, as it imposes that when a fake intersection is found the reward gained has to be lower than inflicted penalty. For instance, to achieve this goal, (P2) can be reformulated as:

$$\max_{X,Y,Z} 2N_{\text{MAX}} \sum_{i=1}^{N} \sigma (\alpha f^*) \prod_{k<j} J_{kj} - (N_{\text{MAX}} - N).$$

where $J_{kj}$ is now made with the substitution (13). Despite aforementioned solution, others can be investigated.

**Numerical evaluation**

In this Section, theoretical results discussed in past Sections are evaluated. In particular, a wide simulation campaign has been carried out to analyze each possible application derived from discussed previous notions. The adopted hardware consists of a desktop computer equipped with Intel i7 970 and 16GB RAM. In all the optimization-based applications, GA implemented in MATLAB R2020a, is employed as SA of reference. For the sake of generality, measurement units are simplified to pure numbers.

**Volumes measure error analysis.** In order to evaluate results obtained in “Computation of volume measure covered by intersecting geometrical loci”, a reference scenario is taken into account. Given the general equation of a circle

$$C(x_0, y_0, r) : (x - x_0)^2 + (y - y_0)^2 - r^2 = 0,$$

the aim is to compute the union area of $C_1 = C(-1, 0, 1)$ and $C_2 = C(1, 0, 1)$, which are external tangent, placed in a mesh 4x2. This task can be done exploiting Eq. (40):

$$\int_{M} \Omega [N(C_1), N(C_2)](x) \, dx =$$

$$\int_{-1}^{1} \int_{-2}^{2} \Theta \left( \alpha \left( -((x + 1)^2 + y^2 - 1) \right) \right) \, dxdy + \Theta \left( -((x - 1)^2 + y^2 - 1) \right) \, dxdy.$$

First, the integral has been calculated in MATLAB through the Riemann Summation, with a mesh precision $\rho = 10^{-3}$. The exact result of the area should be $2\pi R^2 \simeq 6.28318$ but, due to the precision used, the experimental one is $\simeq 6.28313$. This evaluation considers HSFGs that have a discontinuity in 0 that does not represent a problem as stated in Remark 2. However, when SFGs are used, $\alpha$ has an non-negligible impact on the error as discussed in “Computation of volume measure covered by intersecting geometrical loci”.

As expected Fig. 4 shows that, fixed the mesh precision, the error of $\sigma(\cdot)$, with respect to $\Theta(\cdot)$, decreases as $\alpha$ increases. Moreover, the first two orders shown demonstrate that (i) the high-order derivatives accounts for lower $\alpha$ error (ii) for $\alpha \geq 10$ the error is negligible.
Custom volume measure. Thanks to this analysis and results obtained in “Custom D-dimensional volumes”, it is possible to solve problem in the scenario proposed in “Computation of volume measure covered by intersecting geometrical loci”. First, the function that describes the Square #1, namely $S_1$, centered in $[3,3]^T$, can be easily derived from Eq. (50):

$$S_1 = \Lambda_{\alpha}\left[x - 1, -(x - 5), y - 1, -(y - 5)\right].$$

Similarly, it is possible to build the other two squares $S_2$ and $S_3$. The integrand of Eq. (40) is shown in Fig. 5. As previously mentioned, the edges of the volume have a value of 1/2 that is exactly $\sigma(0)$. Finally, exploiting Eq. (40), with $\rho = 10^{-3}$, $\alpha = 10^3$ and a mesh 8x8, the calculated area is 29.99998.

HSF penalty method. To probe results of “HSF Penalty Method” two simulations of different complexity have been done.

The first involves two fixed circles $C_1 = C(-2, 0, 1)$ and $C_2 = C(2, 0, 1)$ placed in a 8x8 mesh. A third one, defined as $C_3 = C(x, y, 1)$ has to be positioned in order to maximize the total covered area. Taking advantage of Eq. (40) and (13), $V_\sigma$ is defined as the SF variant of the Heaviside’s one.

The task is accomplished by the objective function:

$$V_\rho \left[C_n\right] \quad \forall n \in (1, 3),$$

where $\alpha = 10^3$ and $\rho = 5\times10^{-2}$. The locus in which the center $\{x, y\}^T$ of $C_3$ has to be placed to maximize the total covered area is shown in Fig. 6. As expected, the center should not be in proximity of the two pre-positioned circles.

The second experiment emulates a constrained simple scenario in which a drone, equipped with a camera, has to take off and land in the same spot. Throughout the mission, the drone has to pursue a trajectory and scan the maximum possible ground area which is composed of camera shoots over the $(x, y)$-plane. It is assumed that the drone cannot exceed a maximum speed, it flies at constant altitude and the ratio of acquired images is constant and set to unity.
From an optimization point of view, the problem can be stated as follows:

$$
\max_{X,Y} V_\sigma \left[ \{C_n\} \right] \quad \text{s.t.} \\
\left\| [x_k y_k]^T - [x_j y_j]^T \right\| \leq v_{\text{MAX}} \quad \forall k \in (1, N - 1), \\
\quad \forall j \in (k + 1, N), \\
x_1 = y_1 = x_N = y_N,
$$

where $N$ is the number of circles $C_n$, whereas $X$ and $Y$ are the sets of centers $n : 2..N - 1$ coordinates of which trajectory is made of.

The start/end points are settled to $[-2 -2]^T$. Since the acquisition ratio is set to unity, the speed $v$ of the drone corresponds to the distance that is travelled between two consequent points.

Motivated by Eq. (56), it is possible to reformulate the problem stated above as:

$$
\max_{X,Y} V_\sigma \left[ \{C_n\} \right] \prod_{k<j} J_{kj},
$$

where

$$
J_{kj} = \sigma \left( \alpha \left( \left\| [x_k y_k]^T - [x_j y_j]^T \right\| - v_{\text{MAX}} \right) \right).
$$

The test has been run in a $4 \times 4$ mesh with precision $\rho = 10^{-2}, N = 7$ circles, $v_{\text{MAX}} = 1.5$ and $\sigma = 10^2$. Results of the solution of this problem are shown in Fig. 7. Based on solution found by the algorithm, the area covered is $\sim 11.5$. Due to the applied constraint, projected areas are overlapped and cut by the mesh borders. It also results that the adopted speed in each segment is about the maximum possible because, indeed, it minimizes the overlaps.

Custom D-dimensional volumes. In the previous Subsection an interesting pattern has been obtained in terms of covered area. The associated $V_\sigma$ can be useful to test results achieved in “Custom D-dimensional borders”. The outcome of this simulation is shown in Fig. 8, using a $5 \times 5$ mesh, $\rho = 10^{-3}$ and $\sigma = 10^3$. It is worth noting that, even if holes are present in the union of volumes, this is not a problem and a union border $\Gamma$ can always be realized.

Curves intersections. Another aspect of optimization, discussed in “Curves intersections”, is intersection counting and discovering. Two scenarios have been settled. In the first, two circles $C_1 = C(-0.5, 0, 1)$ and $C_2 = C(0.5, 0, 1)$ are considered. In the second, a circle $C_3 = C(0, 0, 1)$ and a triangle $\mathcal{T}$ of vertices ($-0.9, 1.4$), ($1.4, -0.9$), and ($-0.9, -0.9$) are studied. Thanks to formulation in (P3), results of these cases are shown in Figs. 9 and 10 where $N_{\text{MAX}} = 7$ and $\sigma = 10$. In both cases, the returned vector of solution contains the ordered set of information being: the number $N$ of actual intersections (2 and 6 respectively) followed by the related coordinates and then the fake $N_{\text{MAX}} - N$ ones. Experimentally, it is important to mention that as the complexity of the involved scenario increases, the research of intersections gets harder in the sense that some runs may not found all of them. In particular, when two or
Figure 8. Border of area covered by drone's trajectory.

Figure 9. Intersection points of two circles found via GA.

Figure 10. Intersection points of a circle and a triangle found via GA.
more intersection are close, distinguishing becomes an hard process that could be overcome as fine-tolerance algorithms are used. Another essential note is correlated with custom volumes and, specifically, for what concern oblique segments. On a trial basis, outcomes of simulations confirm that ρ related to these pieces of volumes should be set not greater than 10 with the aim of speed up the SA operations.

Conclusion
The present work discussed a mathematical characterization of binary logic granted by the HSF. Then, an analytical representation through logistic function has been studied. Furthermore, the existence of a correlation between the conceived mathematical structure and a finite-time program has been proved.

Based on these concepts, different geometrical applications and a Penalty Method for constrained optimization problems has been deeply investigated. First, the measure of binary operations among D-volumes is presented. These can also be designed in a custom manner so that a compact region is encompassed by a set of loci. Hence, it is even possible to obtain the border analytical expression of the desired D-volume. Finally, a novel Penalty Method has been realized allowing the reformulation of a constrained non-convex optimization problem in an unconstrained one.

A wide simulation campaign demonstrated the effectiveness of proposed methodologies which can be employable in many engineering areas such as computer vision and graphics, medical imaging, robotics and automation. However, several aspects can be further investigated. First of all, a solid correspondence between control structures and proposed HSFGs/SFGs will be addressed. Then, a method to compute the border length of a custom volume will be probed. Current formulation to find intersections could also be improved. Finally, the proposed Penalty Method will be tested with more SAs and compared with other solutions, to settle a proper benchmark.

Received: 23 November 2021; Accepted: 28 April 2022
Published online: 14 May 2022

References

1. Moschovakis, Y. N. What is an algorithm? In Mathematics Unlimited-2001 and Beyond 919–936 (Springer, 2001).
2. Bauer-Mengelberg, S. Review: Kurt godel, b. meltzer, on formally undecidable propositions of principia mathematica and related systems i. J. Symb. Logic 30(3), 359–362 (1965).
3. Turing, A. M. On computable numbers, with an application to the entscheidungsproblem. J. Math 58(345–363), 5 (1936).
4. Savage, J. E. Models of computation: Exploring the Power of Computing Vol. 136 (Addison-Wesley Reading, 1998).
5. Böhm, C. & Jacopini, G. Flow diagrams, turing machines and languages with only two formation rules. Commun. ACM 9(5), 366–371 (1966).
6. Zhou, C., Chen, Z. & Li, M. A parallel method to accelerate spatial operations involving polygon intersections. Int. J. Geogr. Inf. Sci. 32(12), 2402–2426 (2018).
7. Park, S. C. & Shin, H. Polygonal chain intersection. Comput. Graph. 26(2), 341–350 (2002).
8. Chin, C. A. & Wang. Optimal algorithms for the intersection and the minimum distance problems between planar polygons. IEEE Trans. Comput. C 32(12), 1203–1207 (1983).
9. Dobkin, D. P. & Kirkpatrick, D. G. Fast detection of polyhedral intersection. Theor. Comput. Sci. 27(3), 241–253 (1983).
10. Žalik, B. Two efficient algorithms for determining intersection points between simple polygons. Comput. Geosci. 26(2), 137–151 (2000).
11. Martinez, F., Rueda, A. J. & Feito, F. R. A new algorithm for computing boolean operations on polygons. Comput. Geosci. 35(6), 1177–1185 (2009).
12. Gao, Y., Wu, B., Luo, J., Qu, H. Gpu-based arbitrary polygon intersection area algorithm, DEStech Transactions on Engineering and Technology Research (2017).
13. Liu, Y. K., Wang, X. Q., Bao, S. Z., Gomboši, M. & Žalik, B. An algorithm for polygon clipping, and for determining polygon intersections and unions. Comput. Geosci. 33(5), 589–598 (2007).
14. Scheins, J. J., Boschen, F. & Herzog, H. Analytical calculation of volumes-of-intersection for iterative, fully-3-d-pet reconstruction. IEEE Trans. Med. Imaging 25(10), 1363–1369 (2006).
15. C. Liu, S. Yu, M. Yim, A fast configuration space algorithm for variable topology truss modular robots, in: 2020 International Conference on Robotics and Automation (ICRA), Paris, France, 2020.
16. Kim, Y. J., Lin, M. C. & Manocha, D. Collision space 1933–1956 (Springer, 2019).
17. Xu, J., Liu, Z., Yang, C. & Pei, Y. Minimal distance calculation between the industrial robot and its workspace based on circle/polygon-slices representation. Appl. Math. Model. 87, 691–710 (2020).
18. Xu, J., Liu, Z., Yang, C., Li, L. & Pei, Y. A pseudo-distance algorithm for collision detection of manipulators using convex-plane polygons-based representation. Robot. Comput. Integr. Manuf. 66, 101993 (2020).
19. Zhou, B., Chiang, Y.-J. & Yap, C. Soft subdivision motion planning for complex planar robots. Comput. Geom. 92, 101683 (2021).
20. El-Salamaony, M. & Guayl, A. Enhanced modified-polygon method for point-in-polygon problem. In Recent Advances in Engineering Mathematics and Physics (eds Farouk, M. H. & Hassanine, M. A.) 47–61 (Springer International Publishing, 2020).
21. Rao, S. S. Engineering Optimization: Theory and Practice 5th edn. (John Wiley & Sons, 2019).
22. Russell, S. J. & Norvig, P. Artificial Intelligence: A Modern Approach 3rd edn. (Global Edition, Pearson Education, 2016).
23. Holland, J. H. & Reitman, J. S. Cognitive systems based on adaptive algorithms. In Pattern-Directed Induction Systems 313–329 (Elsevier, 1978).
24. Kyurkchiev, N. & Markov, S. Sigmoid Functions: Some Approximation and Modelling Aspects (LAP LAMBERT Academic Publishing, 2015).
25. Iliev, A., Kyurkchiev, N. & Markov, S. On the approximation of the step function by some sigmoid functions. Math. Comput. Simul. 133, 223–234 (2017).
26. Smith, A. E., Cost, D.W. Constraint-handling techniques—penalty functions. Handbook of Evolutionary Computation (1997) C5.2:1–C5.2:6.
27. Kuhn, H.W., Tucker, A.W. Nonlinear programming. Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability (1951).
28. Blum, L., Cucker, F., Shub, M. & Smale, S. Bézout’s Theorem 187–200 (Springer, 1998).
Author contributions
I.G. and I.C. conceived the presented ideas and performed the analytic calculations. I.C. developed the mathematical formalism. I.G. carried out the simulations. All authors discussed the results and contributed to the final manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-022-11941-y.

Correspondence and requests for materials should be addressed to G.I.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. © The Author(s) 2022