WHAT IS HOMOTOPICAL ABOUT HOMOTOPY TYPE THEORY?

CHENTIAN WU

Abstract. In this paper I try to answer the question: *what makes Homotopy Type Theory (HoTT) genuinely homotopic?* By bridging Martin-Löf's dependent type theory and classical homotopy theory, we (1) show how Martin-Löftype theory's core components - universes, dependent types, and identity types - naturally encode geometric concepts when viewed through a homotopical lens, (2) demonstrate how HoTT internalizes five key homotopy-theoretic ideas: paths as equality proofs, fibrations as dependent types, equivalences, univalence, and higher inductive types, and (3) apply these to reconstruct two classical results: the computation of $\pi_1(S^1) \simeq \mathbb{Z}$ via path induction and transport, and the Seifert-van Kampen theorem through pushouts of higher inductive types. The analysis reveals how HoTT captures topological phenomena synthetically, replacing analytic machinery with type-theoretic primitives.

Contents

1. Why Homotopy inside Type Theory?

Homotopy Type Theory (HoTT) merges two seemingly disparate fields: the logical rigor of type theory and the geometric intuition of homotopy theory. To appreciate its significance, we begin by contextualizing its foundations.

Type Theory formalizes mathematics through a computational lens. In this framework, propositions are represented as *types*, and proofs as *terms* inhabiting those types. For instance, the statement "2 is even" corresponds to a type $\mathsf{Even}(2)$, and a proof of this statement is a term $t : \mathsf{Even}(2)$. *Dependent Type Theory* extends this idea by allowing types to depend on terms: if A is a type and $B(x)$ is a type for each $x : A$, the dependent product type $\prod_{x:A} B(x)$ encodes universal quantification ("for all $x : A$, $B(x)$ holds"). This mechanism underpins modern proof assistants like Coq [2] and Agda [1]. However, traditional type theory treats equality simplistically—terms a and b are either judgmentally equal ($a :\equiv b$) or not, discarding any higher-dimensional structure behind why they are equal.

Homotopy Theory, on the other hand, studies spaces through continuous deformations: paths between points, homotopies between paths, and so on. Vladimir Voevodsky's groundbreaking insight was to reinterpret types as spaces, terms as points, and equality proofs $p : a = b$ as paths from a to b. HoTT elevates equality to a first-class geometric notion, preserving not just whether two terms are equal but how they are equal. For example, two programs proven equal in HoTT may follow distinct computational paths, analogous to different routes connecting the same endpoints.

The fusion of these ideas crystallized with Hofmann and Streicher's 1994 groupoid model and Voevodsky's later contributions: the *univalence axiom* (equating equivalent types) and *higher inductive types* [6] (defining spaces by specifying points and paths). Together, they enable HoTT to internalize homotopy-theoretic concepts directly into type theory. For computer scientists, this enriches program verification with geometric reasoning; for mathematicians, it offers a language where proofs are both human-readable and machine-checkable.

2. Martin-Löf's Dependent Type Theory: A Topological Lens

Modern homotopy type theory rests on three structural features of Martin-Löf dependent type theory—*universes*, *dependent types*, and *identity types*. Interpreting each through a geometric lens reveals how the syntax of proofs encodes classical homotopy theoretic ideas.

2.1. **Universes and Type Families as Stratified Spaces.**

To quantify over types without reproducing Russell-style paradoxes Martin-Löf introduced an ascending hierarchy

$$\mathcal{U}_0 : \mathcal{U}_1 : \mathcal{U}_2 : \cdots,$$

where every level $\mathcal{U}_i$ itself lives in the next, a property called *cumulativity.* We can notice the same pattern when we are assembling a CW-complex: the n-skeleton depends only on cells attached in lower dimensions, yet remains a subspace of the completed object. Thus each universe operates like a skeleton that supports all types—and hence all spaces—constructed so far, while leaving room to adjoin further "cells" at the next stage.

Given a base type A a *type family* $B : A \to \mathcal{U}$ assigns to every point $x : A$ a fibre $B(x)$. Topologically one recognises the assignment as a fibration $p : \sum_{x:A} B(x) \to A$ whose total space is the Σ-type $\sum_{x:A} B(x)$ and whose fibre over x is literally $B(x)$. When B is the constant function $\lambda(x : A).\ C$ the fibration becomes trivial, $A \times C \xrightarrow{\pi_1} A$; when B varies non-trivially one obtains twisted bundles, covering spaces, or more exotic constructions such as local systems of higher groupoids.

2.2. **Dependent Types as Continuous Constructions.**

Definition 2.1 (Π-types as Sections)**.** Given a family $B : A \to \mathcal{U}$ the dependent function type $\prod_{(x:A)} B(x)$ consists of terms that pick a point $f(x) : B(x)$ *continuously* in x. Equivalently a term f determines a section $s : A \to \sum_{x:A} B(x)$ via $s(x) :\equiv (x, f(x))$, and the definitional equality $\pi \circ s = \mathsf{Id}_A$ reads "s is a genuine section" after projection.

From a logical standpoint Π-types internalize universal quantification: the judgment $t : \prod_{x:A} B(x)$ is a constructive proof of "for every x we have $B(x)$". From a homotopical standpoint sections detect whether a bundle admits global trivializing data, a theme that resurfaces when Π-types are used to model parallel transport and connection forms.

Definition 2.2 (Σ-types as Total Spaces)**.** Dually, the dependent pair type $\sum_{(x:A)} B(x)$ represents the space obtained by *gluing* all fibres together. A point $(a, b) : \sum_{x:A} B(x)$ simultaneously witnesses the existence of $a : A$ and a dependent element $b : B(a)$; it therefore embodies the constructive content of the existential quantifier. In topology the same ordered pair marks a geometric point lying over a in the total space of the fibration.

Because Σ and Π are adjoint (yes, by this we mean adjunctions in category theory) in an appropriate sense, one already glimpses a categorical formulation of fibre bundles: sections are right adjoints to projection, while total-space formation is left adjoint.

2.3. **Identity Types as Path Spaces.** The conceptual breakthrough of HoTT is to interpret propositional equality as a space of paths in the topological sense. For two terms $a, b : A$ the identity type $\mathsf{Id}_A(a, b)$ carries as inhabitants the paths from a to b; the canonical constructor $\mathtt{refl}_a : \mathsf{Id}_A(a, a)$ is the constant path. Two radically different equalities coexist:

$a \equiv b : A$	$p : \mathsf{Id}_A(a, b)$
judgmental, enforced by definitional computation	propositional, a geometric path that may vary

Judgmental (or definitional) equality corresponds to syntactic normalization, whereas propositional equality admits many distinct witnesses; the latter is where homotopy becomes useful.

2.4. **Path Induction - Contracting the Interval.** Every identity type is governed by the *path induction* (or J) rule: to prove a property C of arbitrary paths it suffices to prove it for the trivial path. Formally, given a family $C : \prod_{x,y:A} \mathsf{Id}_A(x, y) \to \mathcal{U}$ and a term $d : \prod_{x:A} C(x, x, \mathtt{refl}_x)$ there exists a function

$$\mathsf{J}(d) : \textstyle\prod_{x,y:A} \prod_{p:\mathsf{Id}_A(x,y)} C(x, y, p),$$

unique up to definitional equality. From a geometric vantage point path induction says that the unit interval contracts onto a point, so any construction that is homotopy-invariant over the interval already follows from its value at the constant path.

2.5. **A First Glimpse of Homotopy.** Suppose $p, q : \mathsf{Id}_A(a, b)$ are two non-identical proofs of equality. They materialize as distinct paths between a and b. Applying identity types once more, $\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p, q)$ is the type of homotopies $p \Rightarrow q$, and iterating the process generates a tower of higher paths whose globular structure equips every type with an ∞-groupoid of its points, paths, homotopies, and so on. Grothendieck envisioned such structures as the language in which algebraic topology should be carried out; in HoTT they arise automatically from the basic rules of type formation and path induction.

So far, we have now reinterpreted the building blocks of Martin-Löf theory — universes, dependent types, identity types, and path induction — through geometry. Table 1 summarizes the interpretation of Type Theory in Logic and Topology.

Type Theory	**Logic**	**Homotopy Theory**
$A : \mathcal{U}$	a proposition A	a topological space A
$a : A$	a proof a of proposition A	a point a in space A
$A \to B$	$A \Rightarrow B$ (implication)	a continuous map $A \to B$
$P : A \to \mathcal{U}$	a predicate $P(x)$ for each $x : A$	a fibration P over A with fiber $P(x)$ at each $x : A$
$\prod_{x:A} P(x)$	$\forall x : A.P(x)$ (universal quantification)	space of continuous sections of fibration P
$\sum_{x:A} P(x)$	$\exists x : A.P(x)$ (existential quantification)	total space of fibration P
$A \times B$	$A \wedge B$ (conjunction)	the product space $A \times B$
$A + B$	$A \vee B$ (disjunction)	the disjoint union $A \sqcup B$
$p : \mathsf{Id}_A(a, b)$	p is a proof of the equality of a and b	p is a path from a to b in space A
$\mathtt{refl}_a : \mathsf{Id}_A(a, a)$	the reflexivity of equality at a	the constant path at point a
$\mathbb{1}$	`true` (true proposition)	contractible space
$\mathbb{0}$	`false` (false proposition)	empty space
`Prop`	propositions (0-truncated types)	spaces with at most one path between any two points
`Set`	sets (1-truncated types)	spaces with contractible path spaces
n-truncated type	proposition up to level n	space with trivial homotopy groups above level n
function extensionality	functions equal if equal on all inputs	homotopy between functions
univalence	isomorphic types are equal	equivalent spaces are equal
higher inductive type	inductive definition with path constructors	space with specified paths and higher cells

TABLE 1. The HoTT translation dictionary (merged from [5, 7]).

3. KEY HOMOTOPIC INGREDIENTS OF HOTT

In this section we will introduce some classical "*synthetic*"[1] homotopic concepts internalized into HoTT. While there are many more advanced aspects to this topic, to the best of my knowledge from MATH 552, these are the key concepts I've been able to properly understand (given time constraints).

[1] In HoTT, "synthetic" refers to reasoning about spaces and paths directly within type theory without relying on external topological models.

Definition 3.1 (Contractible type)**.** A type C is *contractible* if there exists a point $c_0 : C$ along with a path $\prod_{c:C} \mathsf{Id}_C(c, c_0)$. Geometrically, C resembles a "one-point space" with only higher-order redundancies, and is obviously corresponded with contractible spaces in homotopy.

Definition 3.2 (Equivalence)**.** A map $f : A \to B$ is an *equivalence* when each fiber $\sum_{b:B} \mathsf{Id}_B\,(f(a), b)$ is contractible. Logically, this means "f is both surjective and injective", while from a homotopical perspective, it is called a weak homotopy equivalence.

Definition 3.3 (Higher-inductive type (HIT))**.** HITs extend ordinary inductive types: in addition to point constructors, they allow *path constructors* (and even higher paths) to explicitly generate specified equalities. For example, the circle S^1 is defined by two constructors: $\mathsf{base} : S^1$ and $\mathsf{loop} : \mathsf{Id}_{S^1}(\mathsf{base}, \mathsf{base})$.

Definition 3.4 (Pushout)**.** Given $W \xrightarrow{i} U$, $W \xrightarrow{j} V$, their *pushout* is an HIT with inclusion constructors $\iota_U : U \to \mathsf{PO}$ and $\iota_V : V \to \mathsf{PO}$, plus a path constructor $\alpha : \mathsf{Id}_{\mathsf{PO}}(\iota_U \circ i(w),\ \iota_V \circ j(w))$. This universal property aligns perfectly with topological pushouts in HoTT.

Definition 3.5 (Eilenberg–MacLane space)**.** For an abelian group G and integer $n \geq 1$, $K(G, n)$ is the unique connected space with homotopy group $\pi_n K(G, n) \cong G$ concentrated in degree n. HoTT can represent $K(G, 1)$ using HITs, making its loop space exactly G, with all higher paths trivialized.

3.1. **Paths as Homotopies and Covering Spaces.** Given a type A and points $a, b : A$, the identity type $\mathsf{Paths}_A(a, b) :\equiv \mathsf{Id}_A(a, b)$ is homotopically interpreted as the path space connecting a and b. Iterating identity types yields higher-order paths $\mathsf{Paths}^n_A(a, b)$, whose built-in reflexivity, symmetry, and transitivity laws generate an ∞-groupoid structure—realizing Grothendieck's vision of ∞-groupoids.

The HIT presentation of the circle S^1 – base and loop – ensures that any map from S^1 automatically satisfies the desired "unique recursion centered at the base point" universal property, thereby replacing analytic constructions like CW-structures or piecewise glueing with purely algebraic methods.

For a dependent type $P : A \to \mathcal{U}$, its total space $\sum_{x:A} P(x)$ with projection π forms a *fibration*. *Path induction* (i.e., the J-rule) asserts: to define data dependent on arbitrary paths, it suffices to assign values for the trivial path $\mathtt{refl}_a$. Geometrically, this corresponds to interval contraction, producing a unique and functorial

$$\mathsf{transport}^P_q : P(b_0) \to P(b_1) \qquad (q : \mathsf{Paths}_A(b_0, b_1)),$$

thereby reconstructing the monodromy of covering spaces without explicit "lifting."

3.2. **Π-Types as Sections of Fibrations.** Topology studies continuous sections of a fibration $\pi : E \to A$; type theory's counterpart is the dependent function type $\prod_{x:A} P(x)$. For f inhabiting this type, $s(x) :\equiv (x, f(x))$ defines a section $s : A \to \sum_{x:A} P(x)$, while the judgment $\mathsf{ap}_\pi(f(x)) = \mathtt{refl}_x$ directly translates $\pi \circ s = \mathsf{Id}_A$ into the internal language.

Definition 3.6. If there exists a path $\prod_{x,y:A} \mathsf{Paths}_{P(y)}(\,f(x = y),\ f(y))$ whose projection onto A is $\mathtt{refl}_y$, then f is called *constant up to homotopy.* This is equivalent to $\pi_1(A)$-invariant sections in the classical sense.

Under Voevodsky's *univalence*, pointwise connectedness of two sections implies *judgmental equality*, providing an intrinsic source for function extensionality in HoTT.

3.3. **Equivalences and the Universal Cover.** By Definition 3.2, a map $f : A \to B$ is an equivalence if all its fibers are contractible. The *univalence axiom* further asserts $(A \simeq B) \simeq (A = B)$, elevating equivalences to paths in the universe.

Theorem 3.4. *The loop space ΩS^1 is equivalent to $\mathbb{Z}$: $\Omega S^1 \simeq \mathbb{Z}$.*

Sketch. Define $\mathsf{code} : S^1 \to \mathcal{U}$ with $\mathsf{code}(\mathsf{base}) :\equiv \mathbb{Z}$ and $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}) :\equiv \mathsf{suc}$. The projection of the total space $\sum_{x:S^1} \mathsf{code}(x)$ onto S^1 behaves identically to the classical exponential map $\exp : \mathbb{R} \to S^1$. Contractibility of each fiber implies a bijection between $\mathsf{Paths}_{S^1}(\mathsf{base}, \mathsf{base})$ and $\mathbb{Z}$. □

Without invoking open covers or local trivializations, this example demonstrates how HoTT synthetically reconstructs covering space theory.

3.5. **Lifting Properties and Transport.** Classical HLP asserts: a covering $p : E \to B$ uniquely lifts any homotopy $H : X \times I \to B$. In HoTT, the functoriality and higher coherence of $\mathsf{transport}$ perfectly encapsulate this uniqueness and existence: Given $q : \mathsf{Paths}_B(b_0, b_1)$ and $e_0 : P(b_0)$, we necessarily obtain a unique $\mathsf{transport}^P_q(e_0) : P(b_1)$, with path concatenation corresponding to function composition. Consequently, the entire monodromy action $\pi_1(B) \to \mathsf{Aut}(P(b_0))$ is directly derived from type-theoretic principles.

3.6. **Free Groups and the Seifert–van Kampen Theorem.** SvK computes the fundamental group of a union:

$$\pi_1(U \cup V) \simeq \pi_1(U) *_{\pi_1(U \cap V)} \pi_1(V).$$

HoTT reconstructs this via the pushout HIT from Definition 3.4. Let PO be the pushout of $W \xrightarrow{i} U$, $W \xrightarrow{j} V$; then for any Z,

$$\Big(\prod_{\mathsf{PO}} Z\Big) \simeq \Big(\prod_U Z\Big) \times_{\prod_W Z} \Big(\prod_V Z\Big),$$

where this isomorphism replaces group amalgamation at the section level.

Theorem 1 ($S^1 \vee S^1$)**.** *Generated by point constructors $\mathit{base}_1, \mathit{base}_2$ and path constructors $\mathit{loop}_1, \mathit{loop}_2$, maps into any $K(G, 1)$ (Definition 3.5) depend solely on the images of $\mathit{loop}_1, \mathit{loop}_2$, yielding $\pi_1(S^1 \vee S^1) \simeq \mathbb{Z} * \mathbb{Z}$.*

This "merge-as-pushout" proof can output normal forms via compilers, providing algorithmically meaningful free product reduction.

3.7. **Fundamental Group Action via Deck Transformations.** In the covering $\tilde{X} \xrightarrow{p} X$, $\mathsf{Deck}(\tilde{X})$ is the group of all self-equivalences commuting with p. Encoding the covering as a dependent type $\mathsf{code} : X \to \mathcal{U}$, $\mathsf{Deck}\big(\sum_{x:X} \mathsf{code}(x)\big)$ precisely comprises equivalences preserving the projection.

Theorem 3.8. *For $X = S^1$ and code as in Theorem 3.4, $\mathit{Deck}\big(\sum_{x:S^1} \mathit{code}(x)\big) \simeq \mathbb{Z}$.*

Idea. Each integer $n : \mathbb{Z}$ generates an equivalence $f_n : (x, k) \mapsto (x, k+n)$, while $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^n) = \lambda k.\, k + n$ shows no other equivalences satisfy the commutativity condition. □

Since deck transformations are themselves equivalences, univalence promotes the "$\simeq$" here to a path in $\mathcal{U}$.

4. Semantic Models at a Glance

The consistency of HoTT rests on mathematical models that bridge its syntax with concrete geometric or computational structures. Two models stand out for their philosophical and practical implications.

The Kan Simplicial Set Model [4] interprets types as spaces built from simplices—higher-dimensional analogs of triangles and tetrahedrons. This model validates univalence by treating type equivalences as homotopy equivalences. However, it lacks computational rules for paths: while it guarantees the existence of a path $a = b$, it does not specify how to construct or compute it.

A more constructive approach is offered by Cubical Type Theory [3]. This model introduces an *interval object* $\mathbb{I}$ (representing a continuum from 0 to 1) and *connection operations* to glue higher-dimensional cubes. Here, paths become computable functions over $\mathbb{I}$. For instance, the loop $\mathsf{loop} : S^1$ in Cubical Agda can be evaluated as a concrete sequence of interval manipulations. This model not only justifies univalence constructively but also enables algorithms for normalizing higher-dimensional paths, as demonstrated in our analysis of $\pi_1(S^1)$ in Section 3.

These models are not mere abstractions. They ground HoTT's synthetic reasoning in computation: when we assert $\pi_1(S^1) \simeq \mathbb{Z}$, the cubical model ensures this equivalence is not just symbolic—it reduces to executable code manipulating integers. This synergy between syntax and semantics positions HoTT as both a foundational theory and a practical tool for formal mathematics.

5. Conclusion and Outlook

Homotopy Type Theory redefines the foundations of mathematics by internalizing the geometric intuition of homotopy theory into type theory. We have demonstrated how paths replace equality proofs, enabling synthetic reasoning about spaces like the circle S^1. Dependent types model fibrations, recovering classical covering spaces and monodromy through transport, while univalence simplifies the treatment of equivalent structures. Higher inductive types allow defining spaces synthetically, bypassing cumbersome topological machinery.

These advances are not merely theoretical. The proof of $\pi_1(S^1) = \mathbb{Z}$, once a laborious theorem in algebraic topology, now fits within a few lines of Agda code. Looking ahead, HoTT promises to reshape mathematical practice. Synthetic homology could define homology groups directly via higher inductive types, avoiding simplicial complexes. Formalized algebraic geometry might encode schemes in type theory, enabling verified computations. Pedagogically, HoTT proof assistants could make advanced topology accessible to undergraduates, transforming abstract concepts like fibrations into interactive code.

References

1. Agda Developers, *Agda*.
2. Andrej Bauer, Jason Gross, Peter LeFanu Lumsdaine, Mike Shulman, Matthieu Sozeau, and Bas Spitters, *The hott library: A formalization of homotopy type theory in coq*, 2016.
3. Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg, *Cubical type theory: a constructive interpretation of the univalence axiom*, 2016.
4. Daniel M Kan, *Abstract homotopy*, Proceedings of the National Academy of Sciences **41** (1955), no. 12, 1092–1096.
5. Egbert Rijke, *Introduction to homotopy type theory*, 2022.

6. The Univalent Foundations Program, *Homotopy type theory: Univalent foundations of mathematics*, `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.
7. Yuhang Wei, *Synthetic homotopy theory*, 2024.