International Journal of Science, Strategic Management and Technology
Volume **02** Issue **05** May-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Multimodal Fusion Architecture for Real-Time Task Automation

Mr. Shiva Singh
B.Tech (Information Technology) NIET, Greater Noida

Mr. Abdul Khalid
Assistant Professor (Information Technology) NIET, Greater Noida

https://doi.org/10.55041/ijsmt.v2i5.161

Cite this Article: Singh, S. (2026). Multimodal Fusion Architecture for Real-Time Task Automation. International Journal of Science, Strategic Management and Technology, *02*(05). https://doi.org/10.55041/ijsmt.v2i5.161

***Abstract*—**Real-time task automation has evolved from rule-driven scripts into multimodal decision systems that must jointly interpret text, images, audio, and contextual metadata under strict latency constraints. Despite rapid progress in large lan-guage models, vision-language models, and speech systems, many practical automation pipelines still process modalities indepen-dently, fuse them too late, or rely on brittle prompt templates that amplify upstream noise. This paper presents a multimodal fusion architecture for real-time task automation that aligns text, image, and audio representations into a shared latent space before decoding a response or structured action. The overall pipeline follows an Input → Prompt Generation → Fusion-Aware AI Model → Output → Evaluation → Refinement structure, with a bounded feedback loop that retries low-confidence cases and escalates uncertain outputs to a human operator.The proposed design combines modality-specific encoders, lightweight projection adapters, a shared fusion representa-tion, prompt conditioning, schema-aware output validation, and confidence-based routing. We evaluate the framework on three representative automation tasks: document summarisation with embedded charts, voice-driven form filling, and image-grounded question answering. Compared with a manual baseline, the auto-mated pipeline reduces average task completion time from 142.4 seconds to 31.2 seconds and improves output consistency from 64.0% to 81.3%. The gains are strongest when outputs have a verifiable structure, while tasks requiring open-ended judgement still benefit from human oversight. Beyond empirical results, we discuss robustness issues observed during deployment, including prompt drift, modality mismatch, chart parsing errors, and refinement-loop instability. The paper concludes that practical multimodal automation depends not only on model quality, but also on careful interface design between preprocessing, fusion, validation, and fallback stages.*Index Terms*—multimodal fusion, real-time automation, trans-former, prompt engineering, vision-language models, speech pro-cessing, confidence estimation, task automation

I. Introduction

Automation systems are no longer asked to process only neat, structured inputs. A modern task may arrive as a screen-shot, a short spoken instruction, a partially filled PDF, and a text message that clarifies intent. Traditional automation, built around deterministic scripts and fixed UI actions, per-forms poorly in such settings because the input is ambiguous, cross-modal, and noisy. A practical automation system must therefore do more than classify or transcribe: it must integrate heterogeneous evidence, infer the user's intent, produce an actionable output, and decide when not to act.Recent progress in transformer-based architectures has made this shift feasible. Large language models have shown strong reasoning and instruction-following ability, vision-language models can ground text in images, and weakly supervised speech systems can transcribe a wide range of audio with usable accuracy [1], [2], [4], [5]. Yet deploying these ca-pabilities in a real-time automation stack remains difficult. In many practical systems, each modality is processed by a separate service and their outputs are stitched together only at the end. This late integration is convenient to implement but often loses the cross-modal signals that matter most, such as whether a spoken instruction refers to a region in an image, or whether a chart visually contradicts the textual summary that accompanies it.

The central idea of this work is that real-time automation benefits from a shared multimodal representation rather than a sequence of isolated modality-specific decisions. We there-fore propose a fusion-oriented architecture in which modality encoders feed a common latent space, prompt construction is aware of modality availability and quality, and a bounded refinement loop intercepts low-confidence outputs before exe-cution. The framework is intentionally designed around oper-ational concerns: latency, repeatability, validation, escalation, and robustness to messy real inputs.

A. Background

The broader field of multimodal machine learning covers representation learning, alignment, translation, co-learning, and fusion across modalities [3]. In research, these problems are often studied under clean benchmark conditions. In au-tomation, however, the environment is harsher: inputs may be incomplete, contradictory, poorly formatted, or time-sensitive. A user might speak over background noise while uploading a blurred screenshot of a chart and expecting a structured answer in seconds. This shifts the design priority from pure benchmark accuracy toward stable end-to-end behavior.

A useful observation from practice is that automation quality often depends less on any single model than on the interfaces between system stages. When the text prompt is formed without reference to image availability, the image may effectively be ignored. When speech is transcribed with-out preserving uncertainty, downstream reasoning treats weak evidence as ground truth. When validation is absent, the system may confidently produce malformed or unsafe actions. These failure points motivate an architecture where fusion,

evaluation, and refinement are first-class components rather than afterthoughts.

B. Problem Statement

Existing multimodal automation pipelines suffer from three recurring limitations:

1) **Modality silos:** text, image, and audio are frequently processed in parallel but not jointly represented, which weakens cross-modal reasoning.
2) **Accumulated latency:** chaining multiple services intro-duces serialisation, transport, and orchestration overhead that becomes noticeable in interactive workloads.
3) **Prompt brittleness:** language-model prompts often de-pend on upstream formatting assumptions; small varia-tions in intermediate outputs can cause large shifts in final behavior.

These issues are especially problematic in settings where a system must both answer and act. An automation framework is not only expected to generate plausible text; it must also decide whether the generated result is reliable enough to be executed, retried, or escalated.

C. Objectives

This work has four objectives:

1) Design a multimodal architecture that converts hetero-geneous inputs into a shared representation suitable for downstream reasoning and structured action generation.
2) Maintain latency within an interactive budget suitable for real-time use.
3) Introduce an evaluation-and-refinement loop that im-proves reliability without creating uncontrolled retries.
4) Compare the proposed pipeline against a manual base-line using metrics that reflect practical usability, in-cluding completion time, repeatability, and refinement frequency.

D. Contributions

The main contributions of this paper are:

- A modular multimodal fusion pipeline that aligns text, image, and audio embeddings in a shared latent space before decoding.
- A modality-aware prompt generation strategy that adapts to missing or low-quality inputs.
- A confidence-based routing mechanism that chooses among acceptance, refinement, and human escalation.
- An empirical comparison with a manual workflow across three representative automation tasks.
- A set of deployment-driven observations on robustness, prompt drift, modality mismatch, and operational trade-offs.

II. Literature Review

A. Multimodal Representation Learning

The modern multimodal literature is heavily influenced by large-scale contrastive pretraining. CLIP demonstrated that text and image encoders can be aligned through natural lan-guage supervision at scale, enabling strong zero-shot transfer without task-specific tuning [1]. Its importance lies not only in performance but also in showing that a shared representation space can serve as a practical alignment mechanism. However, CLIP itself is discriminative rather than generative, which limits its direct use in automation pipelines that must produce text, structured actions, or mixed outputs.

The broader survey by Baltrus˘aitis *et al.* formalised multi-modal fusion into early, late, and hybrid strategies, and remains a useful conceptual reference [3]. In production settings, late fusion remains attractive because it is easy to debug and swap components independently. The downside is that late fusion usually collapses each modality into a separate decision before interaction, which sacrifices the interdependence that makes multimodal reasoning valuable.

B. Vision-Language and Audio-Language Systems

Vision-language models such as Flamingo and BLIP-2 extend aligned representation learning toward grounded gener-ation by connecting visual features to frozen language models through learned adapters or querying mechanisms [2], [4]. These architectures are particularly relevant to automation be-cause they preserve the strengths of large pretrained language models while adding visual grounding. Even so, practical systems still face a familiar issue: when image evidence and textual prior disagree, the language prior often dominates, especially for tasks requiring fine-grained chart or document understanding.

On the audio side, large-scale weakly supervised speech recognition has substantially improved robustness across ac-cents and recording conditions, with Whisper-style systems showing strong generalisation in diverse environments [5]. Au-dioPaLM and related work aim to move beyond transcription by bringing audio into a broader language-centric reasoning space [6]. This is promising for automation because spoken instructions often carry timing, emphasis, and context beyond literal transcription. At the same time, clean transcription is not equivalent to correct automation; small recognition errors in names, dates, or entities can propagate into action-level mistakes.

C. Transformer Architectures and Adaptation

The transformer architecture remains the dominant back-bone for both sequence modeling and cross-modal learning [9]. In vision, transformer-based backbones have shown that image patches can be processed in a token-like manner, which makes joint architectures with language components more natural [10]. For practical deployment, however, full end-to-end training of multimodal transformers is often too expensive. Parameter-efficient tuning methods such as LoRA provide a useful compromise by adapting large models with relatively few learned parameters [12]. This principle motivates our use of lightweight adapters and limited trainable cross-attention layers rather than a fully trainable decoder stack.

D. Reasoning, Acting, and Retrieval

Real-time automation sits at the intersection of percep-tion, reasoning, and acting. ReAct-style reasoning-and-acting frameworks highlight the importance of separating internal de-liberation from external tool use and execution [11]. Retrieval-augmented generation similarly shows that grounding model outputs in external evidence can improve factual consistency in text-centric settings [8]. However, extending retrieval and grounding cleanly to non-text modalities remains challenging. An automation system that must jointly inspect an image, parse a spoken request, and fill a structured form requires a fusion mechanism that is tighter than document retrieval alone.

E. Research Gap

Across these strands of literature, three gaps remain es-pecially relevant to practical task automation. First, many systems focus on representation quality but do not discuss bounded real-time execution. Second, evaluation often empha-sizes benchmark accuracy while ignoring latency, repeatability, and escalation behavior. Third, fusion is frequently treated as a one-shot step, whereas practical pipelines benefit from revisiting the prompt or alignment when validation fails. The architecture proposed in this paper is designed explicitly around these missing operational concerns.

III. Problem Formulation

Let an incoming task instance be denoted by

$$X = \{x_t, x_v, x_a, x_m\}, \tag{1}$$

where x_t is textual input, x_v is visual input, x_a is audio input, and x_m is optional metadata such as user ID, task family, or schema definition. Not every modality is present for every task.

The system must produce an output

$$Y = \{y, a, c, r\}, \tag{2}$$

where y is the natural-language response, a is an optional structured action, $c \in [0, 1]$ is a confidence estimate, and $r \in \{\texttt{accept}, \texttt{refine}, \texttt{escalate}\}$ is the routing decision.

The practical objective is not simply to maximize output accuracy. Instead, the system aims to optimize the following deployment-oriented criteria:

$$\max \mathbb{E}[U] = \alpha A + \beta S - \gamma T - \delta E, \tag{3}$$

where A denotes task usefulness or correctness, S denotes consistency across repeated runs, T denotes latency, and E denotes unsafe or incorrect execution risk. The constants $\alpha, \beta, \gamma, \delta$ reflect the relative importance of reliability and speed in interactive automation.

Multimodal Encoding

Each modality is encoded independently:

$$\mathbf{e}_t = f_t(x_t), \tag{4}$$
$$\mathbf{e}_v = f_v(x_v), \tag{5}$$
$$\mathbf{e}_a = f_a(x_a), \tag{6}$$

where f_t, f_v, and f_a denote text, vision, and audio encoders respectively.

To align these into a common space, we apply modality-specific projection adapters:

$$\tilde{\mathbf{e}}_t = \mathbf{W}_t\mathbf{e}_t + \mathbf{b}_t, \tag{7}$$
$$\tilde{\mathbf{e}}_v = \mathbf{W}_v\mathbf{e}_v + \mathbf{b}_v, \tag{8}$$
$$\tilde{\mathbf{e}}_a = \mathbf{W}_a\mathbf{e}_a + \mathbf{b}_a, \tag{9}$$

where each projected embedding lies in a shared d-dimensional space, with $d = 768$ in our implementation.

B. Quality-Aware Fusion

Because modalities may differ in quality or availability, the fusion stage uses adaptive weights rather than simple concatenation. Let the modality quality scores be q_t, q_v, and q_a. Then the normalized fusion weights are

$$\alpha_i = \frac{\exp(q_i/\tau)}{\sum_{j\in\{t,v,a\}} \exp(q_j/\tau)}, \tag{10}$$

where τ is a temperature parameter.

The fused representation is then

$$\mathbf{z} = \alpha_t\tilde{\mathbf{e}}_t + \alpha_v\tilde{\mathbf{e}}_v + \alpha_a\tilde{\mathbf{e}}_a. \tag{11}$$

This design allows the system to down-weight weak evi-dence, such as low-quality audio or visually noisy screenshots, without discarding the modality entirely.

C. Training Objective

The adapters are trained using a combination of contrastive alignment loss and task loss:

$$\mathcal{L} = \lambda_{\text{ctr}}\mathcal{L}_{\text{ctr}} + \lambda_{\text{task}}\mathcal{L}_{\text{task}} + \lambda_{\text{schema}}\mathcal{L}_{\text{schema}}. \tag{12}$$

The contrastive term encourages semantically related cross-modal pairs to cluster in the latent space, while the task term optimizes downstream generation or action prediction. The schema term penalizes structured outputs that violate task-specific constraints.

IV. Proposed Methodology

The proposed pipeline consists of six stages: input normali-sation, prompt generation, multimodal fusion, decoding, eval-uation, and refinement. The design goal is not only accuracy but controlled behavior under real-time constraints.

A. Input Stage

Incoming requests are packaged into a structured envelope that may contain zero or more modalities. The input stage performs lightweight normalisation to reduce avoidable down-stream variance. Audio is converted to 16 kHz mono, images are resized with aspect ratio preserved so the longer side is 512 pixels, and text is stripped of problematic formatting artifacts such as broken Unicode separators or layout-induced line noise. Metadata is also attached at this stage, including task family, expected output type, and schema if applicable.

This step seems mundane, but it directly affects stability. In early versions of the system, silent format mismatches caused degraded performance that was difficult to trace. Making normalisation explicit greatly improved debuggability.

B. Prompt Generation

Prompt generation is conditioned on modality presence, modality quality, and output intent. Rather than using a single universal template, the system selects from a small family of prompt templates and injects modality-aware instructions. For example, when audio is present without accompanying text, the model is first instructed to transcribe before deciding. When an image is present without strong contextual text, the prompt explicitly requests grounded description before answer generation. When a schema is provided, the prompt emphasizes valid structured output.

Let $\pi(\mathsf{X})$ denote the prompt generator. Then

$$p = \pi(\mathsf{X}, \mathbf{z}, s), \tag{13}$$

where s denotes task settings such as output format, refusal policy, and escalation rules. In practice, placing prompt construction before final decoding but after modality inspection produced more grounded outputs than using a fixed decoder prompt.

C. AI Model and Fusion

The projected embeddings are integrated into a shared latent representation, which is supplied to a transformer decoder together with prompt tokens. The decoder itself is frozen or lightly adapted, while the trainable parameters are concentrated in the adapters and small cross-attention blocks. This design reduces training cost and memory footprint while preserving enough flexibility for task-specific alignment.

Although the decoder can in principle accept all modalities through tokenized representations, we found it operationally cleaner to keep modality encoding separate and fuse at a controlled interface. This made latency profiling easier and reduced the risk of prompt changes indirectly altering modality preprocessing behavior.

D. Output Stage

The decoder produces either:

1) a free-form textual response,
2) a structured action specification,
3) or both.

Structured actions are expressed as a schema-constrained JSON object, while free-form responses are treated as natural-language explanations or summaries. Outputs intended for execution are always checked before being accepted by the action layer.

E. Evaluation Stage

The evaluation stage aggregates three fast signals:

1) **Self-consistency score** s_{cons}: obtained by comparing re-peated model outputs on the same input under controlled settings.
2) **Confidence score** s_{conf}: estimated from token-level probabilities or decoder uncertainty proxies.
3) **Schema compliance** $s_{\text{schema}} \in \{0, 1\}$: whether the structured output passes validation.

These are combined into a routing score:

$$s = \lambda_1 s_{\text{cons}} + \lambda_2 s_{\text{conf}} + \lambda_3 s_{\text{schema}}, \tag{14}$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

The final decision is

$$r = \begin{cases} \texttt{accept}, & \text{if } s_{\text{cons}} \geq \tau_{\text{cons}} \wedge s_{\text{conf}} \geq \tau_{\text{conf}} \wedge s_{\text{schema}} = 1 \\ \texttt{refine}, & \text{if output is close to valid but uncertain} \\ \texttt{escalate}, & \text{otherwise.} \end{cases} \tag{15}$$

F. Refinement Stage

If the output fails evaluation but appears recoverable, a bounded refinement stage is triggered. The system rewrites the prompt with an explicit failure reason, missing-field reminder, or conflict note and performs another pass. Refinement is capped at two retries because additional passes showed dimin-ishing returns while increasing latency and unpredictability.

G. End-to-End Procedure

Algorithm 1 summarises the complete inference loop.

V. System Architecture

The architecture shown in Fig. 1 deliberately isolates con-cerns between stages. The fusion block is the main learned multimodal interface, while the decoder is largely frozen. Communication between stages uses an in-memory message format instead of network calls between microservices. This decision was important in practice because early service-oriented prototypes spent too much time serialising tensors and intermediate states, pushing total latency beyond acceptable interactive limits.

A. Stage Replaceability

One practical advantage of the architecture is replaceability. The speech encoder can be upgraded without altering the validator. The prompt templates can be refined without re-training the fusion adapters. The decoder can be swapped for a stronger base model as hardware improves. This modularity is useful not only for research iteration but also for deployment, where model availability, cost, or compliance requirements may change over time.

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **05** May-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Algorithm 1 Real-Time Multimodal Automation Pipeline

Require: Multimodal input X
Ensure: Response y, action a, routing decision r
Normalize text, image, audio, and metadata
Encode available modalities to obtain $\mathbf{e}_t$, $\mathbf{e}_v$, $\mathbf{e}_a$
Project embeddings into shared space and compute fused representation $\mathbf{z}$
Generate modality-aware prompt p
for $k = 0$ to 2 **do**
  Decode output (y, a) conditioned on $(p, \mathbf{z})$
  Compute $s_{\text{cons}}, s_{\text{conf}}, s_{\text{schema}}$
  if all acceptance conditions satisfied **then**
    return $(y, a,$ `accept`$)$
  else if $k < 2$ and output appears recoverable **then**
    Rewrite prompt with failure context
  else
    return $(y, a,$ `escalate`$)$
  end if
end for

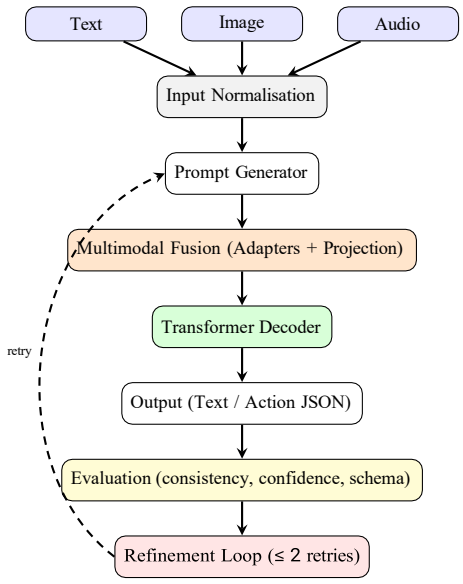

Fig. 1. Multimodal fusion pipeline for real-time task automation.

B. Prompt Before Fusion

A notable design choice was to place prompt generation before final decoding but after modality inspection. Concep-tually, the prompt is treated as a conditioning signal informed by modality presence and task context, rather than as an af-terthought appended to a generic model input. Empirically, this produced more grounded outputs than a fixed prompt attached at the final stage, especially when inputs were incomplete or contradictory.

C. Latency Considerations

Let total latency be approximated as

$$T_{\text{total}} = T_{\text{norm}} + T_{\text{enc}} + T_{\text{prompt}} + T_{\text{fuse}} + T_{\text{dec}} + T_{\text{eval}} + \rho T_{\text{refine}}, \quad (16)$$

where ρ is the expected number of triggered refinements per task. Because ρ is task-dependent, controlling refinement frequency is as important as reducing decoder runtime. This is why the evaluation stage is designed to be lightweight and deterministic.

VI. Experimental Setup

A. Task Families

To evaluate the system across different modality combina-tions, we selected three task families:

1) **Document summarisation with embedded charts** (text + image): The system receives a document or extracted text accompanied by figures, tables, or chart screenshots, and must produce a concise summary.
2) **Voice-driven form filling** (audio + text schema): The system receives spoken instructions and must populate a predefined form or structured field set.
3) **Image-grounded question answering** (image + text question): The system receives an image and a question requiring visual grounding.

Each family contained 50 task instances. The instances were drawn from a mixture of synthetic and anonymised real-world examples to balance coverage and repeatability.

B. Baseline

The manual baseline assumes a human operator with access to standard office tools such as a document viewer, spreadsheet software, a transcription interface, and a general chat-based language model interface, but without integrated multimodal automation. Operators were timed while completing each task from start to finish. A separate reviewer judged output acceptability to reduce bias.

C. Metrics

Four metrics were tracked:

1) **Average completion time (s):** end-to-end task duration.
2) **Iterations to acceptable output:** number of attempts required before the result was judged usable.
3) **Output consistency (%):** agreement across repeated runs of the same task.
4) **Refinement rate (%):** proportion of automated runs that required at least one retry.

We also report failure or escalation rate to reflect how often the system deferred to a human rather than acting automatically.

D. Implementation Details

The automated pipeline ran on a single workstation equipped with a mid-range GPU with 12 GB VRAM. The decoder was a 7B-parameter transformer loaded in 4-bit quan-tised form. Only adapters and a small set of cross-attention parameters were updated. This allowed the system to operate within moderate hardware constraints while retaining enough capacity for cross-modal reasoning.

E. Threshold Calibration

Thresholds were tuned on a held-out set of 20 tasks. We set the consistency threshold to 0.78 and the confidence threshold to 0.65. The schema flag was treated as mandatory for structured outputs. These values are empirical operating points rather than universally optimal choices.

TABLE I
DECISION THRESHOLDS USED IN THE AUTOMATED PIPELINE

Signal	Threshold	Role
Self-consistency score	0.78	Stability gate
Confidence score	0.65	Uncertainty gate
Schema compliance	Pass/Fail	Structural validity
Max refinement retries	2	Latency control

VII. RESULTS AND ANALYSIS

TABLE II
COMPARISON OF MANUAL AND AUTOMATED WORKFLOW

Metric	Manual	Automated
Avg. completion time (s)	142.4	31.2
Avg. iterations to accept	2.6	1.3
Output consistency (%)	64.0	81.3
Refinement rate (%)	N/A	18.7
Failure / escalation rate (%)	4.0	7.3

TABLE III
PER-TASK-FAMILY PERFORMANCE (AUTOMATED PIPELINE)

Task Family	Time (s)	Cons. (%)	Refine (%)
Doc. summarisation	38.5	78.0	22.0
Voice form filling	24.1	86.0	14.0
Image-grounded QA	30.9	80.0	20.0

A. Overall Performance

Table II shows that the automated pipeline reduces average task completion time from 142.4 seconds to 31.2 seconds, corresponding to an approximate 4.56× speedup and a time reduction of about 78.1%. The number of iterations needed to reach an acceptable output also drops from 2.6 to 1.3, indicating that the system usually produces a usable result in one pass and occasionally in two. Output consistency improves from 64.0% to 81.3%, a gain of 17.3 percentage points.

These gains are operationally significant. The time reduction suggests that multimodal automation is most valuable not merely because it is faster than humans at one isolated subtask, but because it reduces tool switching, repeated reformulation, and validation overhead across an entire workflow.

B. Task-Wise Behavior

Table III reveals that performance differs by task family. Voice-driven form filling is the strongest setting: it has the lowest completion time and the highest consistency. This is not surprising. The output schema is known in advance, which makes validation easier, and the system can exploit both transcription and field constraints simultaneously.

Document summarisation with embedded charts is the weakest of the three. It has the highest latency and refinement rate. In our observations, chart-heavy documents are difficult because the visual encoder must interpret small labels, axes, legends, and structured layouts that are unlike ordinary natural images. Errors in chart reading often survive into the summary unless explicitly caught by evaluation or prompt refinement.

Image-grounded question answering falls in between. Man-ual users are already relatively fast at this task, so the speedup is smaller than in form filling, but automation still improves consistency by constraining the response to a stable prompt-and-evaluation regime.

C. Speed-Reliability Trade-off

One notable result is that the automated system has a higher failure or escalation rate (7.3%) than the manual baseline (4.0%). This is not necessarily undesirable. The evaluation stage is intentionally conservative, preferring to escalate uncer-tain outputs rather than silently execute a questionable action. In a real deployment, this means the system acts safely on the majority of routine tasks while reserving ambiguous cases for human review.

D. Operational Interpretation

The results can be interpreted in a practical way:

- **Automation is most effective when the output has structure.**
- **Fusion helps most when modalities complement one another rather than duplicate evidence.**
- **Validation is not optional.** Without it, the speed gains would likely come at the cost of more incorrect actions.

VIII. EXTENDED ANALYSIS

A. Why the Architecture Improves Consistency

Consistency improved by 17.3 percentage points, and this appears to come from three interacting design choices. First, the prompt generator stabilizes instructions based on actual modality availability rather than assuming all inputs exist. Second, fusion occurs before the final response is decoded, which reduces the chance that one modality is treated as secondary context. Third, evaluation rejects unstable out-puts early, preventing low-confidence generations from being counted as final answers.

B. Where the Gains Come From

The measured speedup is not solely the result of faster inference. In the manual workflow, users spend time switching tools, rephrasing queries, checking schema fields, and man-ually reconciling outputs across modalities. The automated pipeline eliminates much of this coordination overhead. In other words, the architecture improves the workflow layer as much as the model layer.

C. Uneven Reliability Across Input Quality

Averaged results conceal an important pattern: reliability is much higher for in-distribution inputs than for noisy edge cases. Long documents, accented or clipped speech, low-light images, and chart screenshots with tiny labels all showed noticeably greater variability. This suggests that deployment planning should not depend only on average-case performance; systems should be calibrated against realistic worst-case modality quality.

IX. Qualitative Error Analysis

TABLE IV
Observed Failure Modes and Likely Causes

Failure Type	Primary Cause	Observed Effect
Chart misreading	Small text / dense lay-out	Wrong trend or axis in-terpretation
Schema omission	Weak field grounding	Missing JSON fields or malformed action
Prompt drift	Inconsistent response structure	Template sensitivity
Audio entity error	Accent / noise / trun-cation	Incorrect names, dates, or values
Cross-modal conflict	Text-image disagreement	Hallucinated explanation
Retry instability	Over-correction during refinement	Different error on each retry

Table IV summarizes the main failure classes observed during development and testing. Several are worth discussing in more detail.

A. Chart and Document Layout Errors

Charts embedded in documents were consistently harder than plain photographs. Small fonts, tight legends, and com-plex axis labeling caused the system to misread trends even when the surrounding text was clear. This failure mode mat-ters because it often produces confident but subtly incorrect summaries rather than obvious nonsense.

B. Speech-Derived Entity Errors

Voice-driven tasks exposed the sensitivity of downstream reasoning to small transcription mistakes. A minor error in a name, form number, or date often propagated directly into the output action. Since many forms require exact values, transcription quality needs to be evaluated not only at the word level but also at the entity and field level.

C. Refinement Instability

The refinement loop helped often enough to justify its inclusion, but it also became a source of hard-to-diagnose behavior. A first retry might fix formatting while breaking factual alignment, and a second retry might restore meaning but violate schema constraints. Limiting the loop to two retries proved to be a pragmatic compromise between recovery and instability.Several observations emerged during day-to-day use that did not fit neatly into the formal metrics but strongly affected system quality.

X. Discussion

The results suggest that multimodal automation is most effective when three conditions hold. First, the task must be decomposable into observable evidence across modalities. Second, the output should have at least partial structure, so that a validator can distinguish acceptable outputs from unsafe ones. Third, the system must be allowed to defer uncertain cases rather than forced to act.

This is why voice-driven form filling performs so well. The schema is fixed, field types are predictable, and the model's job is less about creative generation than about evidence extraction and alignment. Summarisation sits in a middle ground: it benefits from automation but remains harder to validate because many summaries can be syntactically valid while differing in emphasis or completeness. Truly open-ended judgement tasks remain the least convincing domain for full automation. For example, deciding whether a chart is misleading for a particular executive audience is qualitatively different from reading values correctly.

A useful framing, therefore, is not "automation versus humans" but rather "automation with calibrated handoff." The architecture proposed here deliberately supports that framing through escalation logic. The long-term challenge is to make this handoff efficient, informative, and minimally disruptive.

XI. Practical Observations from Real Workflow

A. Prompt Versioning Matters

Prompt iteration turned out to be one of the biggest sources of development effort. Small template edits, such as changing the order in which modalities were introduced or moving the requested output schema higher in the instruction, caused visible changes in model behavior. Maintaining a prompt changelog became necessary to avoid silent regressions.

B. Background Load Affects Repeatability

Some inconsistency that initially looked like model in-stability was eventually traced to inference-time conditions. Background system load influenced response timing and, indi-rectly, output variability. Pinning seeds and lowering decoding temperature improved stability, though sometimes at the cost of reduced flexibility in borderline tasks.

C. Silent Modality Mismatch Is Dangerous

Mismatched sample rates, image resizing artifacts, and truncated uploads rarely caused total failure. Instead, they caused slight degradations that propagated downstream and were harder to catch. The input stage improved substantially after stricter normalisation checks and explicit warnings were added.

D. Retry Caps Improve Predictability

The refinement loop is valuable only when bounded. Be-yond two retries, improvements were rare while latency and unpredictability increased sharply. This simple operational rule prevented the system from spending excessive time trying to rescue cases that should instead be escalated.

XII. Robustness, Safety, and Deployment Considerations

A real-time automation system must be judged not only by average accuracy but by its failure behavior under imperfect inputs and operational constraints.

A. Robustness

Robustness depends on both representation quality and system hygiene. Clean alignment in a shared embedding space helps when one modality is weak, but it does not fully protect against correlated failures. A blurred chart screenshot and a vague textual instruction together can still produce a plausible but wrong summary. This suggests that multimodal systems should explicitly track modality quality rather than treat all available inputs as equally trustworthy.

B. Safety

Safety is especially relevant when outputs can trigger ac-tions. The architecture therefore separates generation from ex-ecution and inserts a validation step between them. Even when the model output appears fluent, an invalid schema, uncertain transcription, or inconsistent second-pass result should block automatic execution. In our design, the system is allowed to be conservative.

C. Deployment

From a deployment perspective, the architecture offers a practical advantage: most of the trainable task-specific logic lives in lightweight adapters and routing rules rather than in the full decoder. This makes it easier to update or retune the system under modest hardware budgets. However, deployment still raises concerns about throughput, queueing, concurrent workloads, and monitoring. A fast single-task latency is not enough if the system becomes unstable under moderate load.

XIII. Ethical and Privacy Considerations

Multimodal automation systems often process sensitive user data including speech, documents, screenshots, and form con-tents. This creates privacy and governance concerns that are easy to underestimate during prototyping.

First, voice inputs may contain personal names, contact de-tails, or background speech from unintended speakers. Second, screenshots and documents may expose confidential business or medical information. Third, automation outputs can create a false impression of certainty because they are fluent and structured. A system that fills a form incorrectly is often more dangerous than a system that refuses to answer.

Accordingly, three design principles are advisable: data minimisation during preprocessing, explicit uncertainty-aware routing before action execution, and auditable logging of why a case was refined or escalated. Although this paper does not present a formal fairness study, the observed sensitivity to accent, image quality, and document format suggests that robustness disparities across user groups deserve careful future attention.

XIV. Limitations

The work has several limitations. First, the evaluation set is relatively small, with 50 instances per task family, and the domain is narrow. Second, the decoder is largely frozen, so any base-model biases or knowledge gaps remain in the pipeline. Third, latency was measured on a single workstation under controlled conditions and may not transfer directly to pro-duction environments with concurrent load. Fourth, although the system supports three major modalities, it does not yet handle video or structured tabular streams natively. Finally, the refinement cap and routing thresholds were tuned empirically and may not generalize across all application domains.

XV. Future Scope

Several extensions appear promising.

A. Learned Prompting

The current prompt generator relies on manually designed templates. A learned prompt policy or routing model could potentially reduce hand-tuning and better adapt to unusual input combinations.

B. Additional Modalities

Structured tables, short videos, temporal event streams, and UI state traces are natural next modalities for automation. Sup-porting them would broaden applicability beyond the current task families.

C. Historical Calibration

The evaluation stage currently treats each task indepen-dently. Incorporating task history, user-specific patterns, or prior correction data could improve calibration and reduce unnecessary escalations.

D. Human-Centered Escalation

Escalation should not merely transfer the raw output to a human operator. A stronger design would include uncertainty explanations, highlighted evidence, and a concise reasoning trace so that the human can intervene efficiently.

E. Cost and Energy Optimization

Running a 7B decoder for every task may be unnecessary. A smaller routing or triage model could handle easy cases, reserving the larger decoder for ambiguous or multimodal-heavy tasks.

XVI. Conclusion

This paper presented a multimodal fusion architecture for real-time task automation built around an Input → Prompt Generation → AI Model → Output → Evaluation → Refine-ment pipeline. The system aligns text, image, and audio signals into a shared latent representation and uses confidence-aware routing to decide whether to accept, refine, or escalate each output. Across three representative task families, the proposed pipeline reduced average completion time from 142.4 seconds to 31.2 seconds and improved output consistency from 64.0% to 81.3%.

The main lesson from this work is that practical multimodal automation depends on more than strong base models. Reliable performance emerges from careful design of modality normal-isation, prompt conditioning, fusion strategy, validation logic, and fallback behavior. The architecture does not eliminate the need for human judgement, especially in open-ended or high-stakes tasks, but it meaningfully reduces routine workload while preserving a mechanism for safe escalation. In that sense, the system is best understood not as a replacement for human operators, but as a structured collaborator that absorbs repetitive multimodal processing and hands back the cases that still require real judgement.

Appendix A Symbol Summary

TABLE V
Main Symbols Used in the Formulation

Symbol	Meaning
X	Input task instance
x_t, x_v, x_a	Text, visual, and audio input x_m
	Metadata / schema / task info $\mathbf{e}_t, \mathbf{e}_v, \mathbf{e}_a$ Raw modality embeddings $\tilde{\mathbf{e}}_t, \tilde{\mathbf{e}}_v, \tilde{\mathbf{e}}_a$ Projected embeddings
$\mathbf{z}$	Shared fused representation
p	Generated prompt
y	Natural-language output
a	Structured action output
c	Confidence estimate
r	Routing decision
s_{cons}	Self-consistency score
s_{conf}	Confidence score
s_{schema}	Schema-compliance score

Appendix B Qualitative Design Sensitivity

Although a full ablation study is left for future work, development observations suggest the following component importance ranking:

1) **Prompt generator:** strongest effect on consistency and grounding.
2) **Schema validator:** strongest effect on safe action gen-eration.
3) **Fusion adapters:** strongest effect on cross-modal align-ment quality.
4) **Refinement loop:** useful for recovery, but only when tightly bounded.

These observations reinforce the idea that practical perfor-mance depends on orchestration and interface design as much as on backbone model capability.

References

[1] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Super-vision," in *Proc. Int. Conf. Machine Learning (ICML)*, vol. 139, 2021, pp. 8748–8763.
[2] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: A Visual Language Model for Few-Shot Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 23716–23736.
[3] T. Baltrus˘aitis, C. Ahuja, and L.-P. Morency, "Multimodal Machine Learning: A Survey and Taxonomy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 423–443, Feb. 2019.
[4] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models," in *Proc. Int. Conf. Machine Learning (ICML)*, 2023, pp. 19730–19742.
[5] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervi-sion," in *Proc. Int. Conf. Machine Learning (ICML)*, 2023, pp. 28492–28518.
[6] P. K. Rubenstein, C. Asawaroengchai, D. D. Nguyen, A. Bapna, Z. Borsos, F. de Chaumont Quitry, P. Chen, D. E. Badawy, W. Han, E. Kharitonov, *et al.*, "AudioPaLM: A Large Language Model That Can Speak and Listen," *arXiv preprint arXiv:2306.12925*, 2023.
[7] W. Chen, X. Ma, X. Wang, and W. W. Cohen, "Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks," *Trans. Machine Learning Research*, 2023.
[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Ku¨ttler, M. Lewis, W. Yih, T. Rockta¨schel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 9459–9474.
[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017, pp. 5998–6008.
[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image Is Worth 16x16 Words: Trans-formers for Image Recognition at Scale," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2021.
[11] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing Reasoning and Acting in Language Models," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2023.
[12] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2022.