



IDC School of Design
अभिकल्प विद्यालय
Indian Institute of Technology Bombay



M.Des Project II

**Redesigning conjunct input for
Swarachakra Kannada**

+

**Empirical study of prediction in
Hindi keyboards**

Prafulla Chandra
18U130022

Prof. Anirudha Joshi
Project Guide

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources.

I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission.

I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.



Prafulla Chandra GS
18u130022

IDC School of Design
Indian Institute of Technology Bombay

Approval Sheet

This Project titled 'Redesigning conjunct input for Swarachakra Kannada & Empirical study of prediction in Hindi keyboards' by Prafulla Chandra GS is approved, in partial fulfilment of the requirements for the degree of Bachelor of Design (Dual Degree Master of Design) at IDC School of Design, Indian Institute of Technology Bombay.

Guide:

Digital Signature
Anirudha N Joshi (i98081)
20-Jul-23 04:42:59 PM

Acknowledgement

I would like to express gratitude to Prof. Anirudha Joshi for his guidance and support which greatly helped start and shape this project.

Thanks to the jury panel for their valuable insights and feedback during the course of the project.

Special thanks to Manjiri Joshi, Rupesh Nath, Ujjwal Jain, Aman Kumar and Narasimhah.

Contents

Introduction	9		
Project Scope	10		
Concept group 1	11		
Separating Vyanjana and Swara operation to two hands / layout groups	11		
1. Primary hand (right) for Vyanjana and Secondary (left) for swara. (hold swara and tap vyanjana)	11		
2. Primary hand (right) for Swara and Secondary (left) for Vyanjana. (tap vyanjana and tap swara)	12		
hold vyanjana and tap swara	12		
Concept group 2	13		
Swipe from swara to swara to form conjuncts.	13		
how to enter chakra / how to add vyanjana?	13		
Option 1. Tap and hold to reveal chakra	14		
Option 2. One hand can be used as a toggle to switch between direct chakra and conjunct mode.	14		
2a. Switch between modes	14		
		2b. Hold on the conjunct bar to activate conjunct mode.	14
		Option 3.	15
		Group 3	16
		Minor UI modifications and Heuristics	16
		I have identified minor UI modifications to optimise the existing swarachakra keyboard for kannada further.	16
		Context: why focus on conjuncts?	17
		About 60% of the words typed through swarachakra have conjuncts	17
		Selected concept	20
		Concept Group 2: Swipe from swara to swara to form conjuncts.	20
		Design	21
		Prototype	26
		Development Brief:	26
			27

Evaluation	28
Empirical evaluation	28
Theoretical evaluation	28
Touch Level Modelling - TLM-GOMS	28
Adopting TLM for Swarachakra	29
Measuring operator times	31
Results	32
Mixer key- Observations	33
Doubler key- Observations	35
Conclusion and Discussion	36
References	37

The project is divided into two parts: a redesign of Swarachakra
Kannada centred on conjunct input and an empirical study of the
use of prediction systems in hindi keyboard.

Part A:

Swarachakra Kannada redesign centred around conjunct input

Introduction

Swarachakra is an Android keyboard developed by IDC School of Design, IITB for Indic Languages. Swarachakra has scope for redesign and further optimisation, especially in inputting conjuncts as they are very common in Kannada.

Swarachakra for Kannada currently exists with the last update being 6 years ago. The keyboard layout is directly adapted from the Hindi version. In this project I'm aiming to modify/redesign the conjunct input technique to observe increased efficiency and reduced learning curve.

If proven to be faster and easier to learn compared to existing design, this project can be taken forward and prototyped for Swarachakra keyboards in other Dravidian languages as they share the same set of conditions.

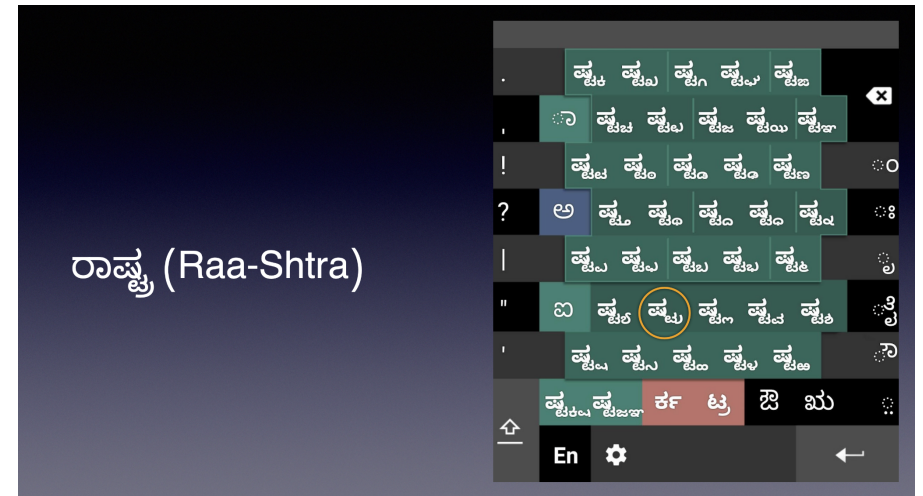
Motivation

I've always wanted to work on a project which would allow me to figure out smaller details of an interaction and optimise it. Working on an existing design like swarachakra and empirically evaluating the iterations would allow me to do so.

Project Scope

Originally I had planned to centre the redesign around conjunct input, chakra layout, swipe gesture based akshara formation etc. But as I understood the usage more, I had to narrow down to conjuncts.

This project focuses on inputting conjuncts using two fingers primarily, using the existing logical (varnamala) layout. Main hand would do the primary task on text input and the secondary hand would work to change interaction mode for the primary hand. The idea is that with familiarisation we can expect the second hand to seamlessly work in switching modes without the user realising it, like pressing shift to enter capital letters.



Concept group 1

Separating Vyanjana and Swara operation to two hands / layout groups

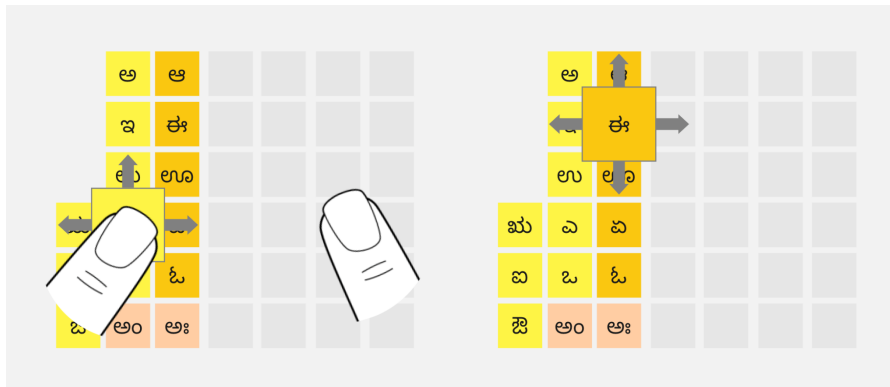


Fig. 1. Layout 1- primary finger on vyanjana.

One hand on selecting vyanjan and other on attaching Swara to it. (In other words one part of layout in on selecting vyanjan and other on attaching Swara)

Currently the finger can cover the chakra while inputting and cause it to select the wrong matra. And left and right fingers have different relative directions they have to move towards for attaching the same swara, ex: left hand has to push away to select “e” while right hand has to pull closer for the same. By separating the two interactions one hand can learn to do

specific tasks better. Ex: left hand can just master in selecting the right Swara and right hand can master in selecting the vyanjan.

And both the parts are always visible in the interface visually ordered according to varnamala, this helps in memorising finger placements. Use one side of the keyboard to enter vyanjana and the other side to attach a Swara to it. The layout can be divided vertically or horizontally like displayed before.

Layout Options:

1. Primary hand (right) for Vyanjana and Secondary (left) for swara. (hold swara and tap vyanjana)

In this option you think of swara to attach first and press vyanjana either simultaneously or after holding. You can retain holding left finger on the swara and move to a different one without lifting and the swara layout is designed to facilitate the same. All the hrusva (single matrakala) swaras; “a” “e” “o” are organised vertically in the order they appear in varnamala and dheerga (extended matrakala) swaras “aa” “ee” “oo” are placed right next to their hrusva variants. Swaras not belonging to these categorisations and yogavahakas are grouped separately.

It might not be the most intuitive thought to type Swara first and Vyanjan later but I am hoping the act of selecting swara fades out and becomes almost a non-cognitive task so it doesn't feel like you are selecting swara first. In this design the secondary finger (left) can participate less by just switching between a relatively smaller number of swaras from an intuitive layout (fig 2) and the primary finger can do the heavy tasks like pointing at the right vyanjana.

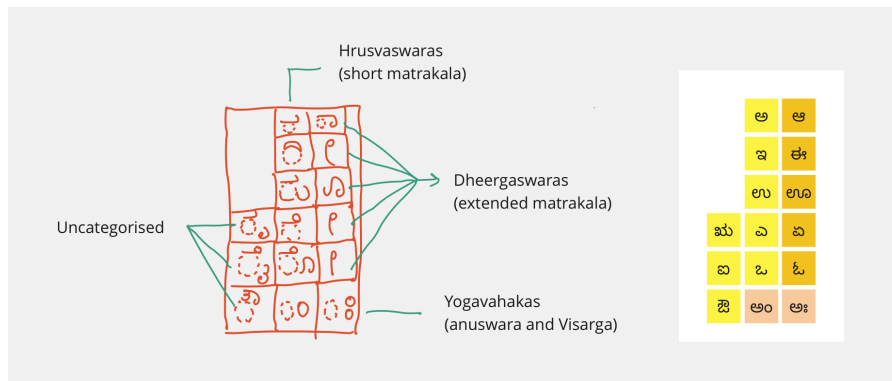


Fig. 2. Swara layout.

2. Primary hand (right) for Swara and Secondary (left) for Vyanjana. (tap vyanjana and tap swara)

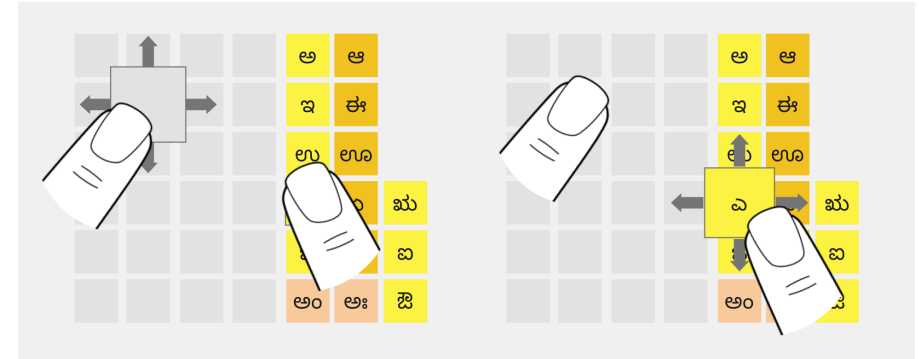


Fig. 3. Layout 2- primary finger on swara.

This design can already facilitate even if you want to enter swara after selecting vyanjan as one can just select vyanjana and use left hand to assign a swara without holding on to it. But to it were to be centred around vyanjana first and swara later the layout can be flipped to visually make it more natural like shown below.

hold vyanjana and tap swara

One more interesting idea would be to make left finger hold and move to change vyanjana and while holding vyanjana swara can be attached just by tapping. This can be accommodate in this layout design

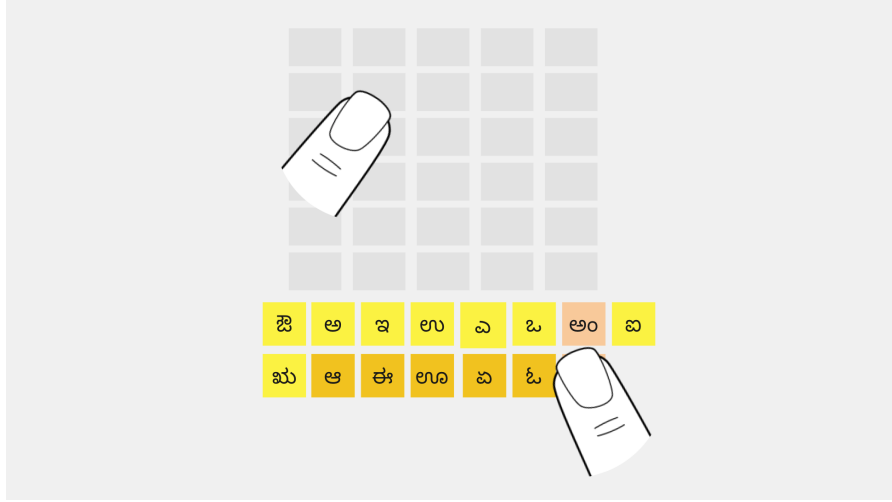


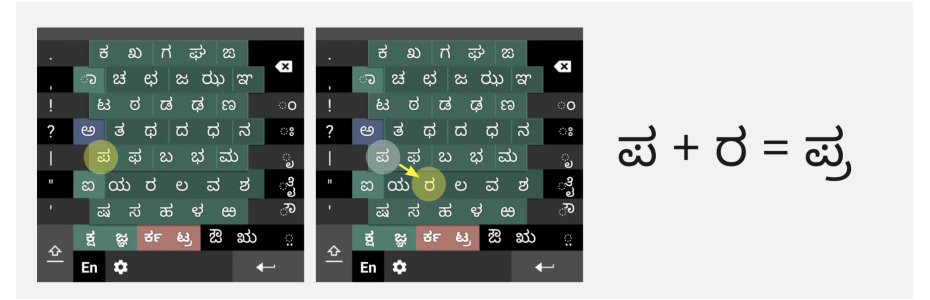
Fig. 4. Vertical layout option.

Other layout options can be explored in this concept like vertical stack as shown above.

Concept group 2

Swipe from swara to swara to form conjuncts.

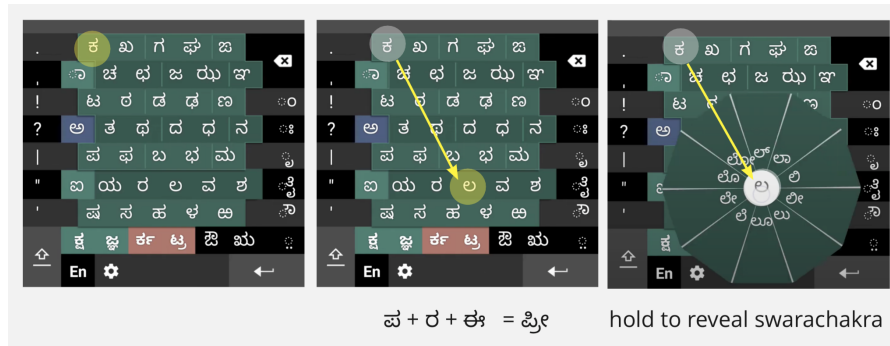
In this design you just swipe from one swara to another to pair them into a conjunct. Attaching a swara to consonant or conjunct can be done in multiple ways as discussed below.



how to enter chakra / how to add vyanjana?

This design poses a new issue, if we can swipe from vyanjan to vyanjan how do you attach a swara, or in other words how do you stop making conjunctions while swiping and instead open swarachakra?

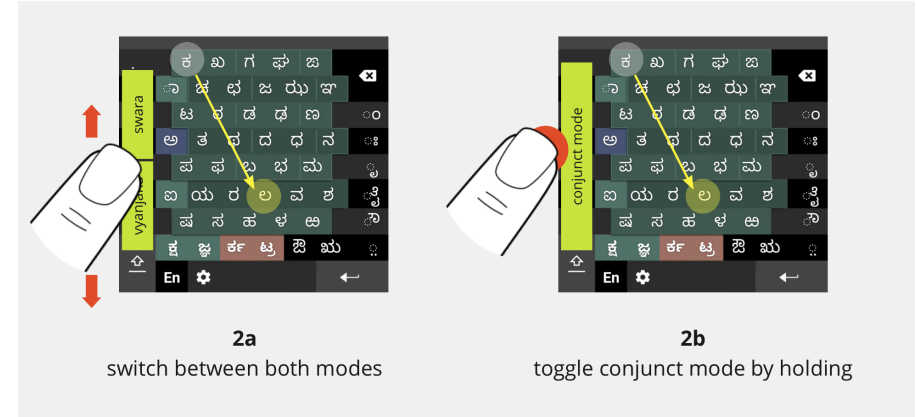
Option 1. Tap and hold to reveal chakra



In this design you just swipe from one swara to another to pair them into a conjunct and keep holding after you combined vyanjanas to open swarachakra and attach swara.

Option 2. One hand can be used as a toggle to switch between direct chakra and conjunct mode.

Based on the frequency of conjuncts this can be done in two ways.



2a. Switch between modes

Switch between modes by moving your finger on a slider type interface. On finer lift it will retain the last selected mode.

2b. Hold on the conjunct bar to activate conjunct mode.

This will let you go from vyanjan to vyanjan to make a conjunct and as soon as you let go it will go to normal swarachakra mode and reveal swara options for selected conjuncts.

Option 3.

We can use the suggested layout from the first concept group to avoid having two different states altogether since both swaras and vyanjanas are always visible together.

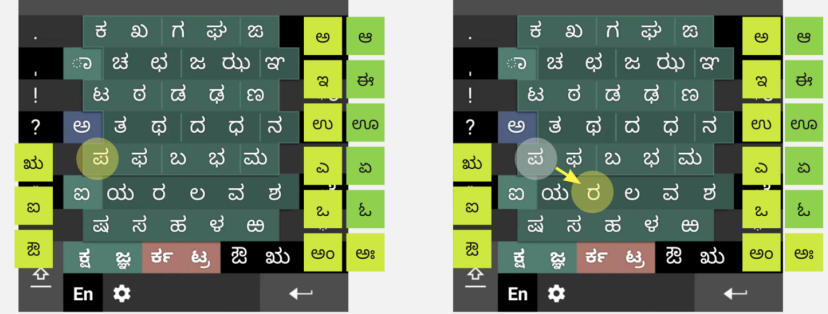
Here are the scenarios explained:



- **Vy - Sw** swipe will make a single conjunct
- **Vy - Vy - Sw** swipe will make a double conjunct
- **Vy tap** will result in **Vy** + "a" matra as default
- **Vy - Vy release** will result in **VyVy** + "a" matra as default
- Tapping and holding **Vy** will make a same class conjunct **Vy_{Vy}**

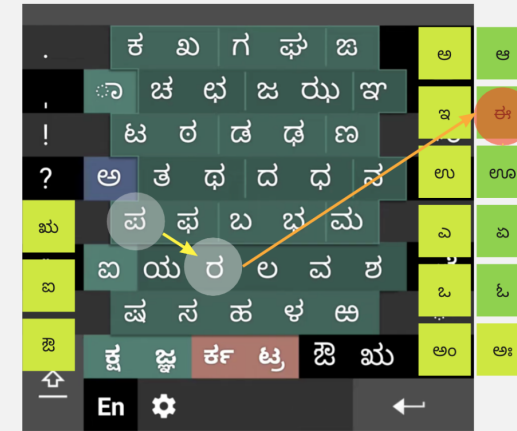
Vy: Vyanjana

Sw: Swara



ಪ + ರ + ಈ = ಪ್ರೀ

pa + ra + ii = prii



Triple swipe

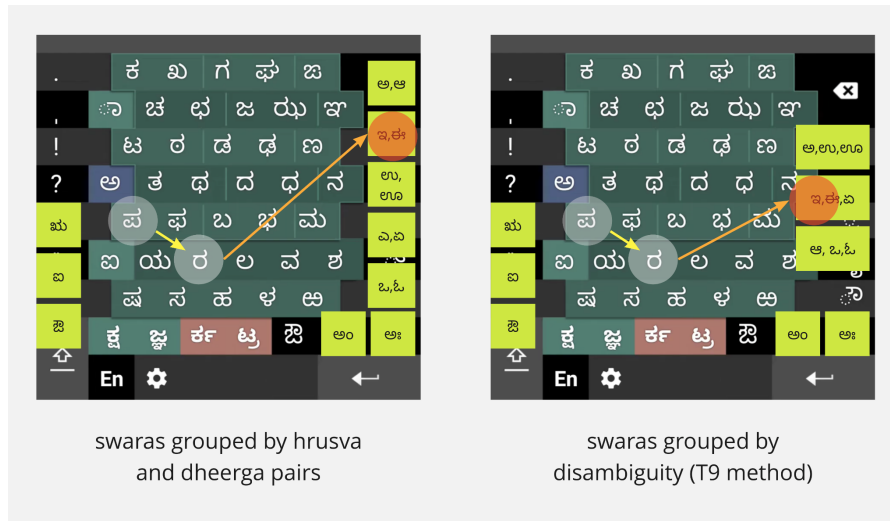
swipe from Vy to Vy to sw

Additionally we can pair multiple swaras into groups in order to reduce added swipe time of third swipe. This can be either 2 swara pairings- “a” and “aa” in one key, which is derived again from varnamala or a three swara combination in unambiguous pairs. The system can disambiguate words using linguistic knowledge (dictionary based- defaults most popular word).

Similar to how T9 keyboards work.

(<https://doi.org/10.1145/332040.332044>- SILFVERBERG, M., MACKENZIE, I. S. AND KORHONEN, P.

Predicting text entry speed on mobile phones)

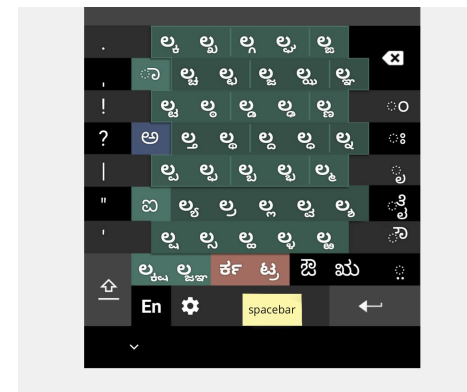


Group 3

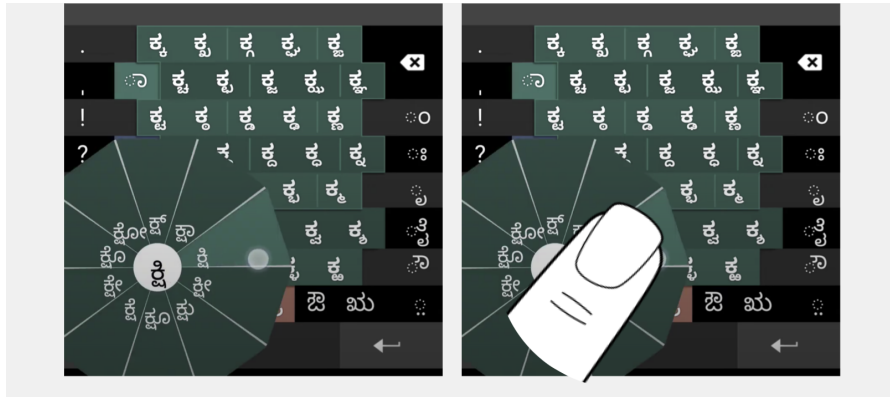
Minor UI modifications and Heuristics

I have identified minor UI modifications to optimise the existing swarachakra keyboard for kannada further.

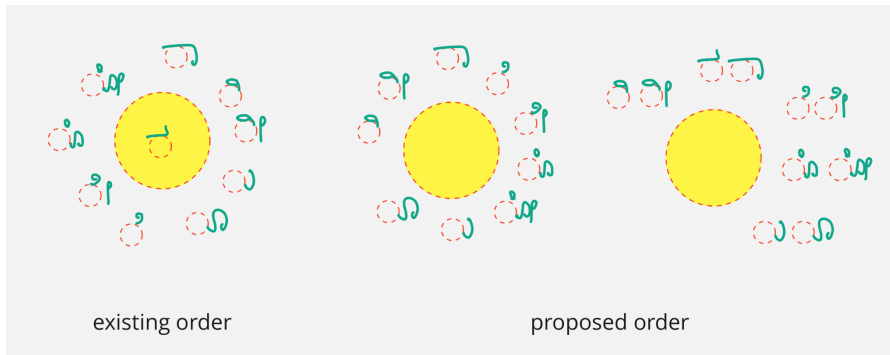
1. Conjuncts don't take appropriate hierarchy in second vyanjan selection mode. After selecting the first vyanjana all the keys show live preview of conjunct which is not helpful as it mainly shows the primary vyanjana making the secondary small. This type of design works for Hindi as conjuncts attach sideways, but has to be improved for Kannada where conjuncts are scaled down and attached to bottom right



2. Finger covers up some of the swaras including the keyboard live preview, users have to just rely on textbox based preview to decide, which is not ideal as most people look at the keyboard while typing naturally.



2. Arrange the swaras in chakra based on how they visually attach to vyanjana deriving from traditional handwriting.



Context: why focus on conjuncts?

About 60% of the words typed through swarachakra have conjuncts

min freq. of words	class	n	% of conjuncts
0	all words	97895	
	conjuncts	60363	61.66%

Tab. 1. Conjunct distribution in Swarachakra corpus.

The occurrence of conjuncts in both written and conversational styles in Kannada is much higher compared to Hindi. Borrowing the design directly from Swarachakra Hindi might not have looked into this scenario.

On analysing all the words typed on swarachakra kannada (data available till june 2015) about 60% of words have conjuncts in them. And the frequency of words with and without conjuncts remain similar, this shows that irrespective of how rare or frequent the words are they have an equal chance of having a conjunct.

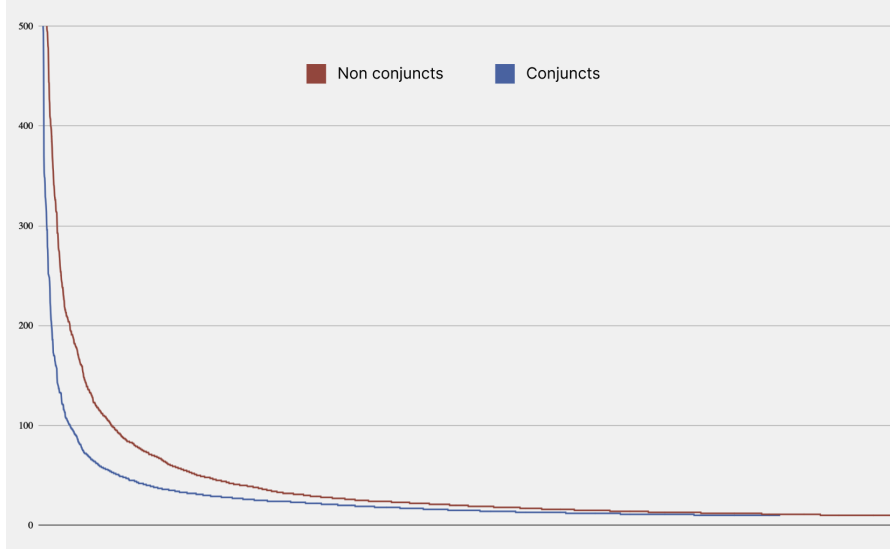


Fig. 1. Frequency of words with and without conjuncts (Swarachakra 2015 Kannada corpus).

Swarachakra corpus is relevant for this analysis as its sourced from text-input compared, and it mainly includes natural conversations. Other corpora ex: Leipzig Corpora Collection: Kannada community corpus (2017) or Wikipedia would include mainly editorial content written by professionals, so they would not reflect real life insights on conjunct frequency.

ವಿಭಕ್ತಿ	ಕಾರಕಾರ್ಥ	ಹೊಸಗನ್ನಡ ಪ್ರತ್ಯಯ	ವರ್ತಮಾನ ಕಾಲ	ಏಕವಚನ	ಕೊಡು+ಉತ್ತ+ಈಯ=ಕೊಡುತ್ತೀಯೆ ಕೊಡು+ಉತ್ತ+ಅನೆ=ಕೊಡುತ್ತಾನೆ(M G) ಕೊಡು+ಉತ್ತ+ಅಳಿ=ಕೊಡುತ್ತಾಳೆ(FG) ಕೊಡು+ಉತ್ತ+ಅದೆ=ಕೊಡುತ್ತದೆ(NG)	Present
ಪ್ರಥಮಾ	ಕರ್ತಾರ್ಥ	ಉ	ವರ್ತಮಾನ ಕಾಲ	ಏಕವಚನ		mādu ut tine
ದ್ವಿತೀಯಾ	ಕರ್ಮಾರ್ಥ	ಅನ್ನು				mādu ut tiye
ತೃತೀಯಾ	ಕರಣಾರ್ಥ	ಇಂದ	ವರ್ತಮಾನ ಕಾಲ	ಏಕವಚನ		mādu ut tale
ಚತುರ್ಥಿ	ಸಂಪ್ರದಾನ	ಗೆ, ಇಗೆ, ಕ್ಕೆ				mādu ut tade
ಪಂಚಮೀ	ಅಪಾದಾನ	ದನೆಯಿಂದ	ವರ್ತಮಾನ ಕಾಲ	ಏಕವಚನ		mādu ut tēve
ಷಷ್ಠೀ	ಸಂಬಂಧ	ಅ				mādu ut tiri
ಸಪ್ತಮೀ	ಅಧಿಕರಣ	ಅಲ್ಲಿ	ವರ್ತಮಾನ ಕಾಲ	ಏಕವಚನ		mādu ut tire
						mādu ut tave

Fig. 1. Left:Vibhakthi pratyaya, Right: Present tense rule.

Moreover the whole idea of framing words in kannada involves agglutinating prefixes and suffixes, which creates conjuncts very often.

Certain grammatical rules also ensure high use of conjuncts, ex: Present tense requires use of “uttha” in the verb form, word suffixes (vibhakthi pratyayas) also dictate use of certain conjuncts [ref. Fig 1]. Even going from written style to spoken style retains the conjuncts in most cases.

min freq. of words	class	n	% of conjuncts	% wrt conjuncts
25	top 2000	2000		
	conjuncts	850	42.50%	
	same class conjuncts	303	15.15%	35.65%
45	top 1000	1000		
	conjuncts	353	35.30%	
	same class conjuncts	150	15.00%	42.49%
80	top 500	500		
	conjuncts	165	33.00%	
	same class conjuncts	76	15.20%	46.06%
300	top 100	100		
	conjuncts	31	31.00%	
	same class conjuncts	17	17.00%	54.84%

Tab. 2 Conjunct distribution in Swarachakra corpus of top 2000, 1000, 500 and 100 words.

One more insight was, even among conjuncts, almost 40% of them were of the same class. Most action verbs use same class conjuncts.

(same class conjunct is where both the consonants combined are same, this is used to increase emphasis of a consonant in pronunciation)

Different class conjunct / Vijathi: Ex:

“gra” = “g+r+a” (grameena)

“kri” = “k+r+i” (kride)

Same class conjunct / Sajathi: Ex:

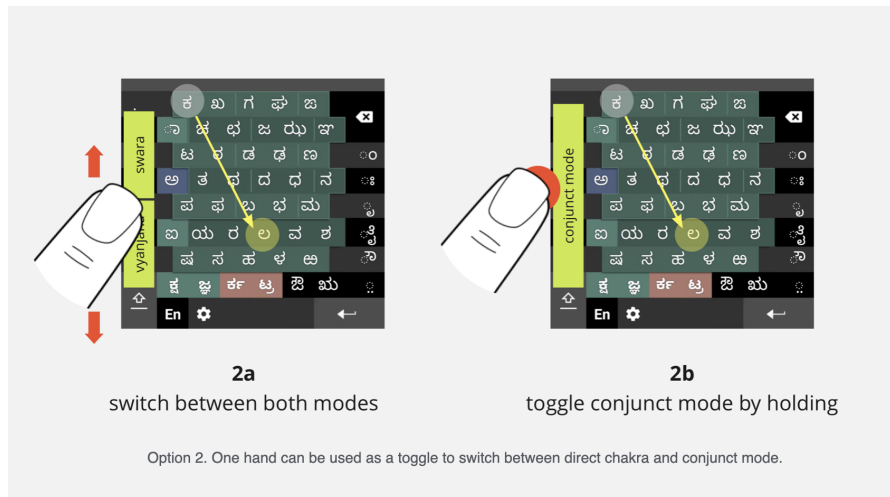
“tta” = “t+t+a” (barutta)

“ddu” = “d+d+u” (bandiddu)

“lla” = “l+l+a” (baralla, illa, hogalla)

Selected concept

Concept Group 2: Swipe from swara to swara to form conjuncts.



Two finger simultaneous input created confusion in the input model as it offered a lot of ambiguity, entering swara or vyanjana first?, and which finger to use primarily?, which to hold and move & which to tap?, hence it was not taken forward.

Moreover selected concept is easier to implement, as its a simple modification of existing layout and would be easier to evaluate against original design.

This concept was selected as it retains the main chakra interaction of swarachakra, and uses varnamala layout (more recognisable).

Design

Animations of the concept can be found here:

https://drive.google.com/drive/folders/1FWML_3oIgHj7HNOL8IhUNJgteFQ2YJ-O?usp=share_link

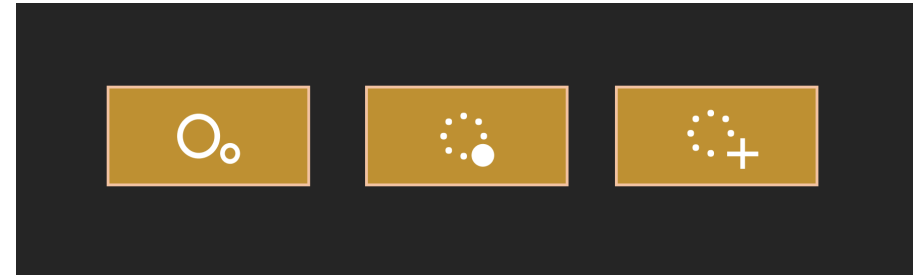
The design adds two new keys to the swarachakra keyboard:

1. Mixer key



The mixer key enables conjunct compose mode, and you can swipe between two consonants to create a conjunct. And then attach the swara if required.

2. Doubler key



doubler key makes it easy to input same class conjuncts. You can use it like caps lock, click on it to engage and the keyboard will show preview of doubled conjuncts and you can just add them just like normally.

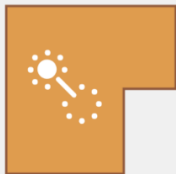
3. Pop up - preview



This also changes default live preview behaviour of swarachakra to replace with a follow-finger pop up preview of conjuncts.

Mixer Key

✦ Mixer



Swipe between Vy and Vy
to form a conjunct

Tap to turn on/off
(Turns off after one use)

or

Hold to turn on
Release to turn off

Doubler Key

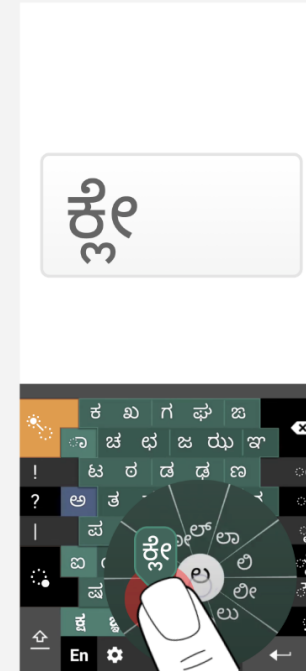
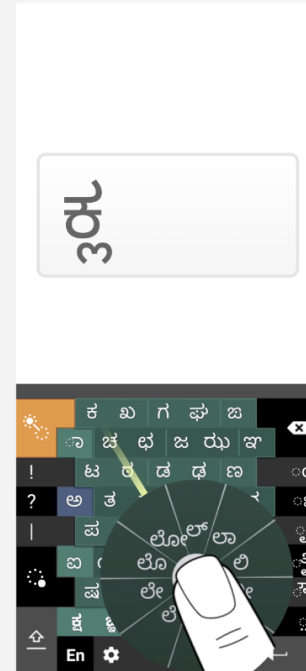
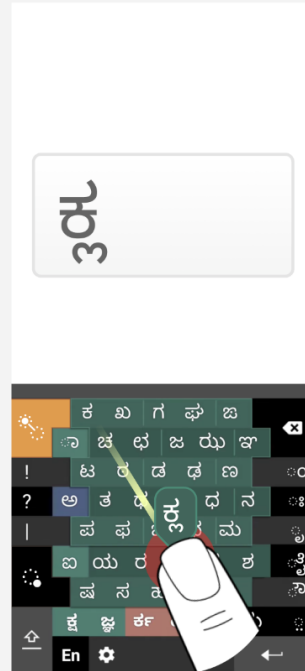
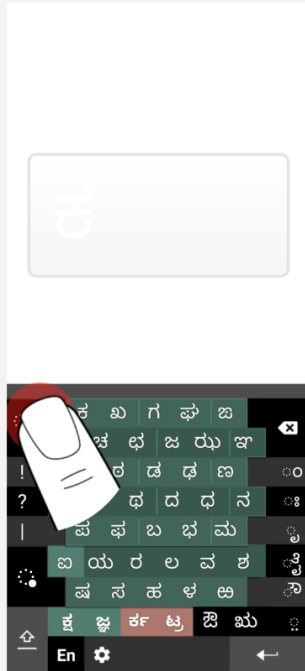
✦ Doubler



Tap on a Vyanjana to input
VyVy- same class conjunct

Tap to turn on/off
(Turns off after one use)

Single hand



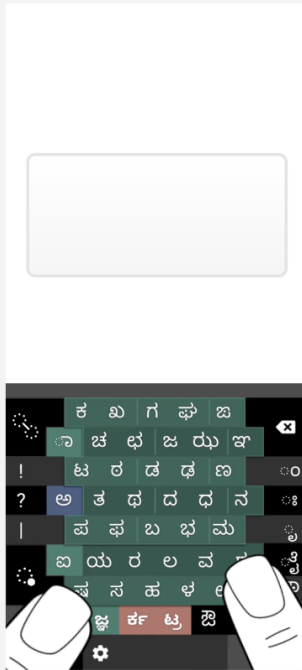
tap on mixer key to
engage conjunct mode

can also be held using
secondary hand

swipe to compose
conjunct and release
finger to reveal chakra

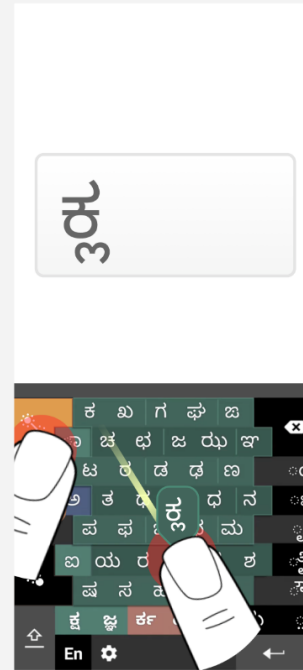
attach swara by tapping
on the wheel

Two hand

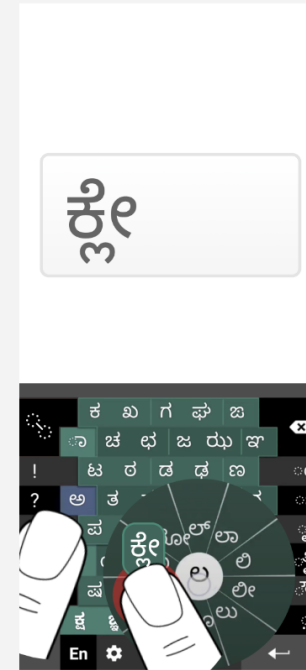


use secondary hand to
engage “mixer” key

can either tap to engage or
tap and hold to sustain



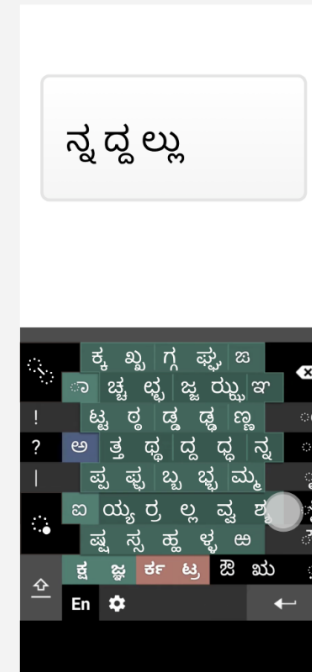
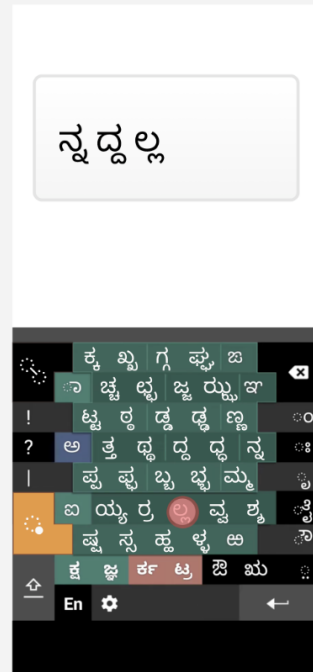
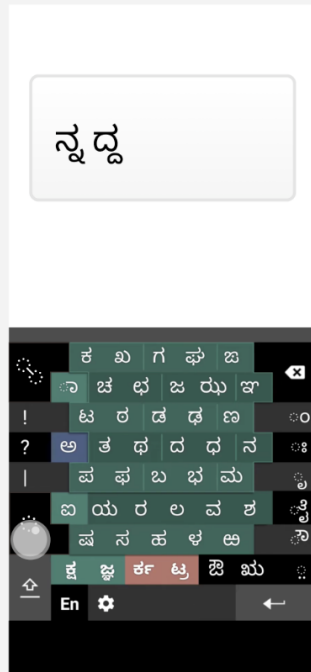
use primary hand to
compose conjunct by
swiping on two
consonants



lift off secondary hand to
reveal the usual swara chakra

use primary hand to
attach swara by tapping

Doubler - One hand



tap on doubler key to
change layout

can also be held using
secondary hand

tap and swipe as usual
but this time it inputs a
same-class conjunct

Change in interaction with single hand mixer mode

While using the mixer key with a single hand, the chakra opens automatically upon finger release after combining consonants. This is contrary to normal behaviour where swara opens on finger touch down. However since the starting points are different between normal and mixer mode, and you are still composing a full conjunct including a swara in this case, the user would understand the difference. Tapping a swara in mixer mode ends the conjunct compose mode and you can start swiping on keys to open chakra like usual after that.

Here the chakra automatically opening upon finger release works like a prompt asking if you need to attach any swara to it, you can dismiss by swiping elsewhere.

The pop up preview complements the swipe and release interaction of mixer mode. During conjunct input eyes will be on finding the vyanjana, so keys can show vyanjana as is to help with primary objective and pop up can show preview to compliment character pointing.

Doubler aims to simplify the mental model of inputting the same class conjunct. Currently you have to swipe twice on the same key. same-class conjuncts are used to emphasise pronunciation of the conjunct, this mode helps them to enter

a doubled conjunct in one go. They can swipe on keys like usual to attach swara.

Prototype (Difficulties)

Development Brief:

priority	design	details	notes
1	Mixer Switch: Swipe to input conjuncts	Swipe on two keys to enter first unicode, conjunct unicode and second unicode	first=touchdown second= u0CCD third=liftoff
2	Doubler Key: Conjunct Lock button to input same-class conjuncts	Engage lock and tap	inserts touchdown unicode + u0CCD and the same first unicode disengages conjunct lock button

3	Swara chakra order modification	change the order of swaras in chakra open view	using new order based on how the swaras attach visually to the consonant rather than existing varnamala order
3	Pop-up Preview	Add a pop-up to show live preview of conjunct	inserts touchdown unicode + u0CCD and live key unicode

Evaluation objectives:

- **Two hand input in conjunct input rate**
Check if the secondary hand adapts to the mixer key, and the friction of using the secondary hand reduces over time and becomes unnoticed.
- **Single hand input in conjunct input rate**
- **Same-class conjunct input rate**

Evaluation

Empirical evaluation

One of the main outcomes of the project was a working android prototype to conduct empirical evaluation with, I hit a major roadblock at the very end of the project timeline. The development was delayed majorly due to developer constraints, by the time I appointed a new developer to work on this, we noticed the code was not the latest one, after sourcing the new code we started to have bigger issues in development.

Even though the logic was figured out, the source code was using deprecated dependencies and the app was not building. We gave up the effort after multiple tries at getting the base app built. On the hindsight I admit this should have been found earlier so project direction could've been changed.

In order to compensate for the learning outcome for myself, I am supplementing this project by continuing an existing empirical study under my project guide. The study is between Swarachakra and Google Indic in Hindi.

Theoretical evaluation

Touch Level Modelling - TLM-GOMS

GOMS is a specialised human information processor model for human-computer interaction observation that describes a user's cognitive structure on four components. (Card et al. 1968)

As explained by Hochstein, the GOMS model consists of Methods that are used to achieve Goals. A Method is a sequential list of Operators that the user performs and Operators are the elementary perceptual, motor or cognitive actions ex: tapping, swiping, scanning.

KLM-GOMS is a keystroke level modelling originally developed for computers in the 90s, TLM- Touch level modelling (Rice et al. 2014) builds on KLM and derives more variables and operators to accommodate modern touch based devices. I have used TLM and modified it to accommodate operators specific to swarachakra interactions.

TLM does not predict the method, instead it evaluates a selected method and predicts time for it. It lists all the operators required to achieve a task and sums up the time.

Hence it does not have broader/abstract goals or method selection rules.

Adopting TLM for Swarachakra

Swarachakra has specific operators and TLM only measures one hand interaction. In order to accommodate two hand interaction, I have added rules and additional operators.

Operators

	operator	ID	definition	Measured time (s)	Proposed time (TLM)
1	homing	h	returning fingers to home location. usually on letters 'va/ॐ' and 'Ta/ॐ'	0.18	-
2	tap	t	tapping on a target on screen	0.35	0.1
3	scan and tap	s	look for a specific target and tap	1.20	1.35

4	scan unique preview layout and tap	su	look for a specific target in unique preview layout* and tap	1.80	-
5	scan doubler layout and tap	sd	look for a specific target in doubler layout* and tap	1.2	-
6	scan special keys and tap	ss	look for a special target* and tap	0.46	-
7	chakra select	c	Select a swara attachment in opened chakra	0.80	-
8	chakra swipe up/down	cu	Select the 'ॐ' ardh akshara (halant) attachment* in opened chakra	0.42	-
9	swipe conjunct	z	Scan and swipe from a swara to another swara*	2.00	-
10	tap and hold	th	Tap and hold on a target	0.00	-

4* - **Unique layout:** existing swarachakra replaces the key layout with a live preview of composed conjuncts during

conjunct input. This makes a unique visual layout for each vyanjan and hence scanning time will be different from normal scan.

5* - **Doubler layout:** this is the proposed layout for doubler key action. It turns the vyanjans into a same-class conjuncts of themselves, effectively making a new visual layout but still maintains the logical order. Hence the times are assumed to be same as normal scan and tap.

6* - **Special targets** refer to non-akshara keys like shift, doubler, mixer key, symbols key etc. These keys are separate class and users memorise the location of them much better than input keys as they have different visual treatments and are kept outside the main varnamala area.

8* - **The '◌ᳵ' ardh akshara (halant)** is an attachment used to write ardh akshara (half-letter) or to prepare the akshara to take another vyanjan turning into a conjunct. This operation is used to create conjuncts. Swiping up on a key is a distinct operation compared to swiping and selecting one out of 10 different attachments. Hence this operation is different compared to normal chakra select.

9* - During the **swipe operation**, the user scans for a target vyanjana and starts the swipe from vyanjana 'A' to 'B'. The

swiping and scanning for the second vyanjana happens simultaneously. Hence the observed times are lower than the sums.

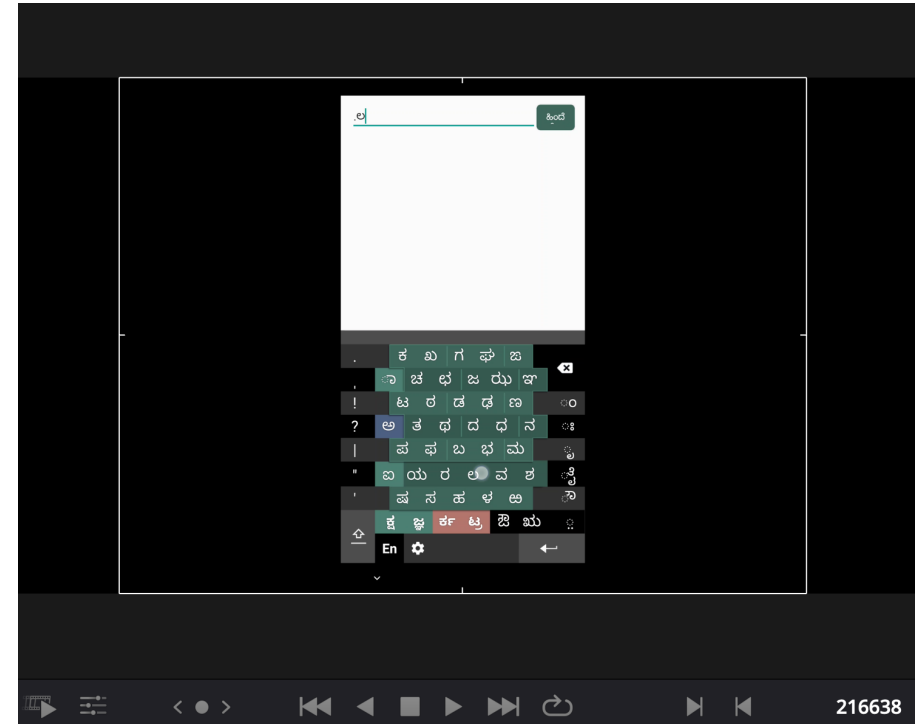
We also notice differences in times among operators 2 and 3, this could be because:

In **2. Tap**, the TLM proposed estimate is made for generic tap, but for swarachakra which has way more touch targets (each key is treated as a target), hence we measure longer tapping times.

In **3. Scan and tap**, even though swarachakra has more targets to scan through, it is laid out in a logical order, which groups in rows of similar vyanjanas, including dheerga and hrusca pairs. Moreover the layout gets familiar after a period of time hence we observe faster times.

Measuring operator times

Frame start	Frame End	Difference	in seconds
217148	217167	19	0.32
217235	217242	7	0.12
217321	217325	4	0.07
216618	216633	15	0.25
216813	216825	12	0.20
216902	216909	7	0.12
216966	216973	7	0.12
217035	217047	12	0.20
217152	217170	18	0.30
217236	217247	11	0.18
217322	217325	3	0.05
217387	217400	13	0.22
217462	217476	14	0.23
217534	217546	12	0.20
217615	217628	13	0.22
217695	217703	8	0.13
			0.18
homing			$h = 0.18 \text{ s}$



Operators were measured by analysing screen recordings and calculating time based on frames. Concepts were shown as real-size screenshots for timing.

Results

Mixer key

type "ಶುಭರಾತ್ರಿ"								
Original swarachakra, two/one hand			Mixer swarachakra, one hand			Mixer swarachakra, two hand		
Operator		time (s)	Operator		time (s)	Operator		time (s)
scan and tap	s	1.20	scan and tap	s	1.20	scan and tap	s	1.20
chakra select	c	0.80	chakra select	c	0.80	chakra select	c	0.80
scan and tap	s	1.20	scan and tap	s	1.20	scan and tap	s	1.20
scan and tap	s	1.20	scan and tap	s	1.20	scan and tap	s	1.20
chakra select	c	0.80	chakra select	c	0.80	chakra select	c	0.80
scan and tap	s	1.20	scan special	ss	0.46	scan special	ss (left)	0.46 (0)
chakra swipe up	cu	0.42	swipe conjunct	z	2.00	swipe conjunct	z (right)	2.00
scan unique	su	1.80	chakra select	c	0.80	chakra select	c	0.80
chakra select	c	0.80	tap	t	0.35			
		9.42			8.81			8.00
Empirical measurement: 8.9s								

Observations

- Swiping from vyanjana to vyanjana is more efficient, it takes less time than its parts (scan+tap+scan+tap). This might be because as the user taps on the first vyanjan, he can directly start swiping without the obstruction of chakra. And during the swipe, the finger is working as a cursor so it might be accelerating scanning.
- Scanning a new unique layout for every halant combination adds too much time to the process.
- Even if we add a generous delay of 1s of two hand delay as realistically both hands don't start the operations at the same time, two hand mixer still outperforms the original design.

Results

Doubler key

type "ಹುಟ್ಟುಹಬ್ಬದ"								
Original swarachakra, two/one hand			Doubler swarachakra, one hand			Doubler swarachakra, two hand		
Operator		time (s)	Operator		time	Operator		time
scan and tap	s	1.20	scan and tap	s	1.20	scan and tap	s	1.20
chakra select	c	0.80	chakra select	c	0.80	chakra select	c	0.80
scan and tap	s	1.20	scan special	ss	0.46	scan special	ss (left)	0.46 (0)
chakra swipe up	cu	1.20	scan doubler	sd	1.20	scan doubler	sd (right)	1.20
chakra select	c	0.80	chakra select	c	0.80	chakra select	c	0.80
scan and tap	s	1.20	scan and tap	s	1.20	scan and tap	s	1.20
scan and tap	s	1.20	scan special	ss	0.46	scan special	ss (left)	0.46 (0)
chakra swipe up	cu	1.20	scan and tap	s	1.20	scan and tap	s (right)	1.20
tap	t	0.35	scan and tap	s	1.20	scan and tap	s	1.20
scan and tap	s	1.20						
		10.35			8.52			7.60
Empirical measurement: 10.03s								

Observations

- Having one separate streamlined interaction for doubling vyanjanas gives a conjunct effectively at the rate of normal scan and tap.
- During inputting the same class conjunct, there is no need to scan again as your finger will be on the same vyanjan after adding the halant, but visually the layout changes to the unique one. This might have a cognitive effect but is not measured in this example.

Conclusion and Discussion

I understand that the theoretical evaluation using GOMS is not the best measure to evaluate the design, but it still shows a significant difference. One of the learning outcomes for me was to practise empirical modelling and conduct the study, and unfortunately I could not do it in this part of the project.

I am supplementing my project with a different empirical study within the domain of text-input (part 2). It is an empirical study investigating the use of word prediction systems between Swarachakra Hindi and Google Keyboard Hindi (Gboard). This is part of ongoing text-input studies in IDC, so I have used corpus material from previous studies and took help from Manjiri Joshi, Rupesh Nath and Ujjwal Jain in planning the study.

Part B:

Does prediction really help in Hindi Text Input?

Empirical pilot study between Swarachakra Hindi (no prediction) and Google Keyboard Hindi (prediction) comparing typing speed and error rates.

This study is a part of ongoing IDC research work on the usefulness of prediction systems in Indic text input. A pilot study of 4 users (70 sessions) is presented as a supplement to my Mdes project.

Study Objective

Does prediction help users input text in Hindi?

Previous studies in Marathi show that prediction-based keyboards have slower typing speeds than keyboards without a prediction system. This could be due to the complex linguistic morphology of agglutinative languages like Marathi, as discussed in the study. The study's motivation was to revisit this question using a modern prediction system and compare Hindi keyboards because Hindi has a low to none level of agglutination.

We compared Swarachakra to Google Keyboard, which intelligently predicts upcoming or currently typing words and displays up to three words in a bar-based interface.

Previous research has shown that participants improve rapidly in the first three hours of typing practice. As a result, we decided to conduct a more practical study. In this study, we present results from a longitudinal evaluation lasting 5-6 hours of practice on two keyboards.

Method

This was a within subject study with 14 sessions per user spread over 7 days. In each session the user participates in a phrase transcription task, using an IDC developed web-app which records the data. Each session consists of 12 phrases of varying difficulty (3-8 words per phrase). The user completes the typing test with one keyboard, waits 15-20 minutes, and then repeats the process with the other keyboard. A round is completed after two sessions. There are seven rounds in total, with the first 6 being for practice in order to observe advanced typing behaviour from participants. The users are presented with the same set of phrases in multiple rounds to accelerate learning. First three rounds with one practice set and last three with a different. After six practice sessions, the users complete a main task typing a completely new phrase set, which will be our primary point of analysis.

User group





























The proposed user group is a subset of tech-savvy individuals between the ages of 18 and 25. The group is made up of Hindi-speaking college and university students who are well-versed in technology, regularly use smartphones, and have never used Swarachakra or Google Indic before.

Because they have already attained a certain level of technological proficiency, these users are likely to be more efficient and effective in text-input. This saves time spent on technology education.

Recruitment

We used convenience sampling to select participants who were available and willing to commit to the entire 7-day study. All the students were from IIT Bombay. Although it is not a perfect representation of the audience, it was done because of time restrictions.

Variables and Counterbalance

Users	Practice phase I			Practice phase II			Main
QWERT							
ASDFG							
ZXCVB							
YUIOP							



Practice sets 1 & 2



Swarachakra



Google Keyboard



Main set

The keyboard order in which the user participates and the practice phrase set were counterbalanced independent variables. As a result, half of the users used phrase set 1 for the first half of practise sessions, followed by phrase set 2. The other half is the opposite. Within these groups, half used swarachakra first and then Google, while the other half did the opposite.

Expected dependent variables were typing speed in cpm (characters per minute), error rate in percentage.

Age, gender, time of study, order of phrases within a phrase set were randomised.

Corpus

The phrases used in both the practise and main sessions are of the same overall difficulty. They are a mix of informal and formal phrases from school textbooks, popular folk songs, children's songs, and well-known quotes. The phrase length ranged from 3 to 8 words (9 to 27 Unicode characters), with 5.5 words or 18.3 Unicode characters being the average.

Practice set 1	Practice set 2	Main set
सत्कर्मों की तूलिका से जीवन में रंग भरों	दोस्त दोस्त ना रहा	हमें अपने दोस्तों से मदद माँगनी पड़ेगी
युद्ध किसी भी समस्या का स्थाई हल नहीं है	इसका क्या अचार डालोगे	राज्यों में विकास को लेकर टक्कर होने लगी
मेरी तबियत खराब है	चढ़े तवे पर रोटी सभी डाल लेते हैं	ईंट का जवाब पत्थर से देना
मेरे पहुंचने से पहले गाड़ी जा चुकी थी	काला अक्षर भैंस बराबर	डूबते को तिनके का सहारा
करारा जवाब देना	बेनीमाधव सिंह गौरीपुर गाँव के जमींदार थे	तुम कल कहाँ गए थे
मेरे पास माँ है	किसी अपठित पाठ्यांश का श्रुतलेखन करो	क्या आप मेरी मदद कर सकते हैं
अक्लमंद को इशारा काफी है	आदमी का सबसे बड़ा दुश्मन गुरूर है	चलो स्कूल चलें
परिश्रमी व्यक्ति अवश्य सफल होता है	हम अगले हफ्ते फिर फ़िल्म देखेंगे	कट्टप्पा ने बाहुबली को क्यों मारा
उसने पशुचिकित्सालय में वैद्यों को बुलाया	मुख्य दूसरों पर हँसते हैं	ज़िन्दगी लम्बी नहीं बड़ी होनी चाहिए
अंधी गली के मुहाने पर अंधी सरकार	राजनीति में कोई पूर्ण विराम नहीं होता	सड़क मरम्मत के लिए बंद है
खाना खाया क्या	हाथ पसारना	कथनी से करनी भली
खूब लाभ होना	मुझे जाना है	कंगाली में आटा गीला

Session plan

		starting keyboard and phrase set			
round	session s	SWC	SWC	GI	GI
1	1,2	Practice set 1	Practice set 2	Practice set 1	Practice set 2
2	3,4	Practice set 1	Practice set 2	Practice set 1	Practice set 2
3	5,6	Practice set 1	Practice set 2	Practice set 1	Practice set 2
4	7,8	Practice set 2	Practice set 1	Practice set 2	Practice set 1
5	9,10	Practice set 2	Practice set 1	Practice set 2	Practice set 1
6	11,12	Practice set 2	Practice set 1	Practice set 2	Practice set 1
7	13,14	Main set	Main set	Main set	Main set
Users		U1	U2	U3	U4
		U5	U6	U7	U8
		U9	U10	U11	UN...

Session schedule per user

USER 1				
phase	round	sessions	phrase set	keyboard
Practice Phase 1	1	1	Practice set 1	SWC
		2	Practice set 1	GI
	2	3	Practice set 1	SWC
		4	Practice set 1	GI
	3	5	Practice set 1	SWC
		6	Practice set 1	GI
Practice Phase 2	4	7	Practice set 2	SWC
		8	Practice set 2	GI
	5	9	Practice set 2	SWC
		10	Practice set 2	GI
	6	11	Practice set 2	SWC
		12	Practice set 2	GI
Main	7	13	Main set	SWC
		14	Main set	GI

SWC= Swarachakra

GI= Google keyboard

Setup

All tests were carried out on a "Oneplus 6" smartphone with a 6.8-inch display and a resolution of 2280 1080 pixels. We used a custom web app that displayed the phrases in accordance with the protocol, captured detailed user logs, and managed user records. The app data for the prediction enabled Google keyboard was cleared after each session.

Results

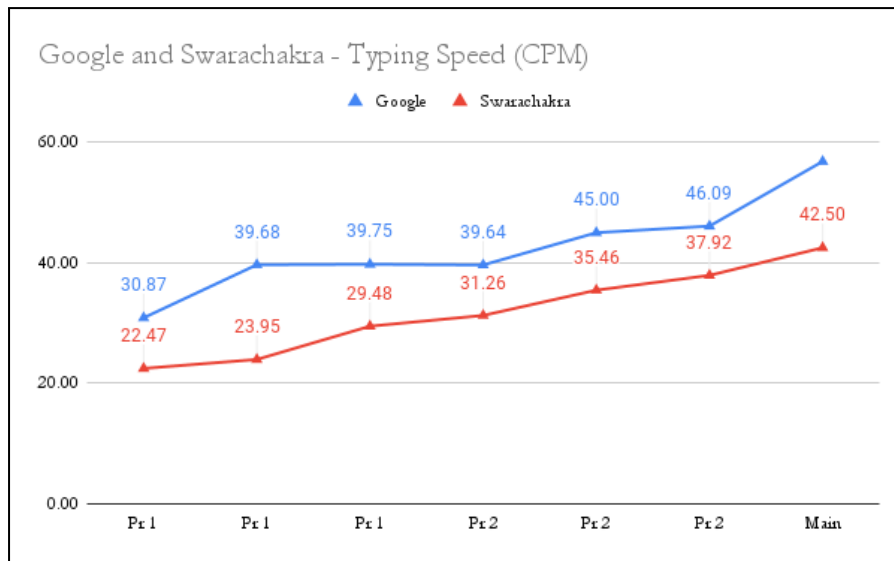
Swarachakra:

The mean typing speed was found to be 31.86 characters per minute (CPM) with a standard deviation of 7.3. The error rate observed for Swarachakra was 2.06%.

Google Keyboard:

The mean typing speed was observed to be 42.55 CPM with a standard deviation of 7.99. The error rate for Google Keyboard was 1.65%.

To determine the statistical significance of the difference in typing speeds between the two keyboards, an independent samples t-test was conducted. The t-test yielded a t-value of -4.093 and a p-value of 0.0001.



For Swarachakra, the 95% CI is approximately (28.798, 34.922). This means that, with 95% confidence, the true population mean typing speed for Swarachakra lies within this interval.

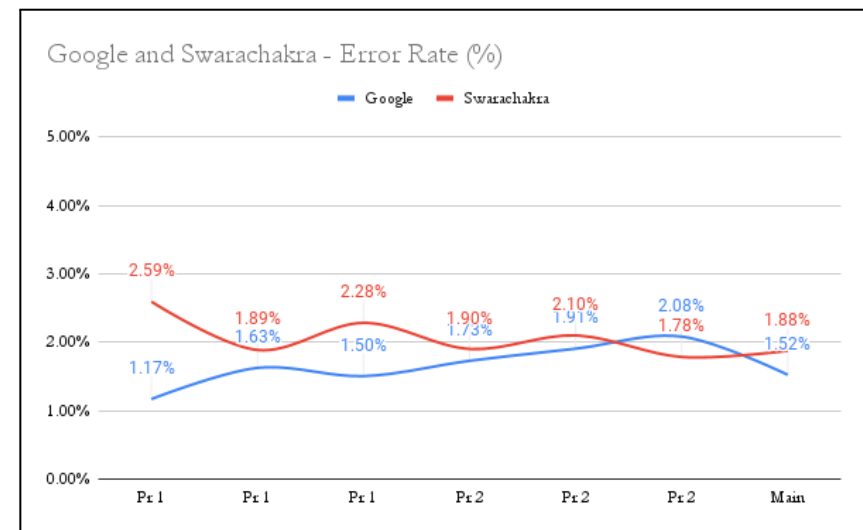
For Google Keyboard, the 95% CI is approximately (39.826, 45.274). This indicates that the true population mean typing speed for Google Keyboard falls within this interval with 95% confidence.

The results of the t-test and the associated 95% CIs indicate a statistically significant difference in typing speeds between

Swarachakra and Google Keyboard. Google Keyboard demonstrated a noticeable higher mean typing speed compared to Swarachakra.

The overall typing speed, measured in characters per minute (CPM), was approximately 10% better with Google Keyboard compared to Swarachakra. Google Keyboard demonstrated higher CPM values throughout the study. However, it should be noted that both keyboards showed consistent growth in typing speed across the rounds, with significant improvements observed when there was a change in the practice phrase set.

Error Rate:



The error rates for both keyboards were similar and remained consistently low, with less than 2.5% errors recorded most of the time. Both Swarachakra and Google Keyboard exhibited comparable accuracy in terms of error rates.

In addition to the quantitative results, several qualitative findings were observed during the study, providing further insights into the performance of Swarachakra and Google Keyboard in Hindi text input:

Phrase-Specific Differences: The study identified certain phrases where the performance difference between Swarachakra and Google Keyboard was particularly notable. For example, the phrase "मुझे जाना है" (I need to go) exhibited a significant difference in typing approach. Users were observed to type only the first word and rely on the prediction feature to complete the last two words. This behaviour was more prominent with Google Keyboard, indicating the effectiveness of its prediction system for completing predictable phrases.

Grammatical Predictability: Certain sentences performed better using Google Keyboard. This could be ascribed to the sentence structure's grammatical predictability. Users could input only a piece of the phrase and then rely on Google

Keyboard's accurate predictions to complete the remaining words.

Influence of Training Data: It was observed that some phrases, such as "परिश्रमी व्यक्ति अवश्य सफल होता है" (Hardworking individuals are certain to succeed) and "अक्लमंद को इशारा काफी है" (A wise person requires only a hint) performed better with Google Keyboard. This could be related to the inclusion of common quotes and proverbs in the original training set used for text prediction in Google Keyboard.

Overall, the study demonstrated that Google Keyboard with prediction yielded higher typing speeds compared to Swarachakra without prediction. Individual user preferences and experience with each keyboard, however, should also be taken into account as potential influences on typing speed and accuracy. Additionally, the error rates were comparable between the two keyboards, showing that prediction in this study did not significantly affect error rates.

Shortfalls

One of the main limitations is the small sample size of only 4 users. With a total of 14 sessions per user, this resulted in a relatively limited dataset of 56 sessions. The small sample size may restrict the generalizability of the results and introduce potential biases.

Secondly, the study employed an accelerated learning approach with 14 sessions per user (7 per keyboard), which may not have allowed sufficient time for participants to fully adapt and improve their typing skills. The relatively short duration of the study might have hindered the participants' ability to reach a stable performance level.

Furthermore, the counterbalancing approach used in the study aimed to minimise order effects and bias by alternating the order of keyboards and practice sets. However, due to the nature of the accelerated learning study design, where participants had limited exposure to each keyboard, the counterbalancing between keyboards within sessions might not have been as effective. A more appropriate approach would have been to counterbalance between keyboards across practice phases sessions, reducing potential confounding factors and providing a more accurate comparison between the keyboards.

Moreover, treating the keyboards separately in future studies would be beneficial. Conducting sessions where one keyboard is used exclusively, followed by sessions where the other keyboard is used, would allow for a more focused analysis of each keyboard's performance. This approach would provide clearer insights into the individual strengths and weaknesses of each keyboard without potential interference from alternating between them.

Conclusion

This study compared the typing speeds and error rates between Swarachakra and Google Keyboard in Hindi text input. The findings revealed that Google Keyboard had significantly higher typing speeds compared to Swarachakra, while the error rates did not show a significant difference.

The study contributes to ongoing research on prediction systems in Indic text input. It highlights the advantage of incorporating a prediction system, as seen in Google Keyboard, to enhance typing efficiency in Hindi. The quantitative analysis, along with the 95% confidence intervals, provides reliable estimates of the typing speeds.

Qualitative observations identified phrase-specific differences and the influence of grammatical predictability and training data on performance. These insights deepen the understanding of user behaviour and the effectiveness of prediction-based keyboards.

However, the study had limitations, including a small sample size, limited duration for accelerated learning, inefficient counterbalancing, and the need to consider external factors. Future research should explore a comparative study between

Hindi and Marathi, accounting for differences in agglutination levels.

While this study serves as a valuable pilot contribution, it cannot be treated as a standalone study due to its supplementary nature.

Uninterpreted data and more graphs including individual, phrase wise analysis etc can be found here:

 [SWCvsGI - Prafulla](#)

References

1. Dalvi, G. *et al.* (2015) "A protocol to evaluate virtual keyboards for Indian languages," *Proceedings of the 7th International Conference on HCI, IndiaHCI 2015* [Preprint]. Available at: <https://doi.org/10.1145/2835966.2835970>.
2. Dalvi, G. *et al.* (2016) "Does prediction really help in marathi text input?," *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* [Preprint]. Available at: <https://doi.org/10.1145/2935334.2935366>.
3. Goms (2022) *Wikipedia*. Wikimedia Foundation. Available at: <https://en.wikipedia.org/wiki/GOMS>
4. *Goms and keystroke-level model* (no date). Available at: https://www.cefns.nau.edu/~edo/Classes/CS477_WWW/Docs/TechArticles/GOMS%20and%20Keystroke-Level%20Model.pdf
5. Rice, A.D. and Lartigue, J.W. (2014) "Touch-level model (TLM)," *Proceedings of the 2014 ACM Southeast*
6. *Regional Conference* [Preprint]. Available at: <https://doi.org/10.1145/2638404.2638532>.
7. Prafulla Chandra [Stage 1 Presentation- M.Des Project II.](#)
8. *Kannada Grammar- tense* (no date) *Kannada*. Available at: <http://www.languagesgulper.com/eng/Kannada.html> (Accessed: October 31, 2022).
9. N, D. and Kumar. P, R. (2012) "Sentence boundary detection in Kannada language," *International Journal of Computer Applications*, 39(9), pp. Table 2. Available at: <https://doi.org/10.5120/4852-7124>.
10. Mind, T. (2021) ಕನ್ನಡ ವ್ಯಾಕರಣ - ವಿಭಕ್ತಿ ಪ್ರತ್ಯಯಗಳು, *The MindPalace Academy of Learning*. Available at: <https://themindpalace.in/index.php/2021/08/16/kannada-grammar-vibhakti-pratyayagalu/>
11. *Multilingual phrase finder*. Available at: <https://omniglot.com/language/phrases/phrasefinder.php?lang2=kannada+kannada+phrases+list>