Razvoj pametnega prezračevalnega sistema

Matic Smogavec, Mitja Truntič, Timi Karner
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko
Koroška cesta 46, 2000 Maribor
E-pošta: matic.smogavec@student.um.si

Development of smart ventilating system

The project focuses on the development of a local ventilation unit with optimized geometry and autonomous operation for more efficient and adaptive air exchange. The hardware is based on an ESP32 microcontroller, a BLDC motor, and Bosch BME680 sensors. The working methods include microcontroller programming, mobile application development, CAD modeling, 3D printing, and airflow simulations. Key improvements include the integration of air profiling sensors in conjunction with algorithm design for adaptive operation, as well as the optimization of the fan, heat exchanger, and the geometry of other components, aiming to reduce energy losses and enhance performance.

Kratek pregled prispevka

Projekt se osredotoča na razvoj lokalne prezračevalne enote z optimizirano geometrijo in avtonomnim delovanjem za učinkovitejše in prilagojeno prezračevanje. Strojna oprema temelji na ESP32 mikrokrmilniku, BLDC motorju in senzorjih Bosch BME680. Metode dela vključujejo programiranje mikrokrmilnikov, razvoj mobilne aplikacije, CAD-modeliranje, 3D-tisk in simulacije pretoka zraka. Ključne izboljšave vključujejo vgradnjo senzorjev za profiliranje zraka v povezavi z načrtovanjem algoritmov za prilagojeno delovanje ter optimizacijo ventilatorja, prenosnika toplote in geometrije ostalih komponent, s ciljem zmanjšanja energetskih izgub in izboljšanja zmogljivosti.

1 Uvod

Razvoj človeštva temelji na izboljševanju življenjskih pogojev, pri čemer ima ključno vlogo bivalna klima, ki vpliva na naše počutje. Ta zajema hrupnost, temperaturo, vlažnost, zračni tlak in sestavo zraka (organske spojine, trde delce, pline). V notranjih prostorih se koncentracija teh snovi neizogibno povečuje, zato je prezračevanje nujno. Ročno prezračevanje z odpiranjem oken povzroča toplotne izgube, zato vse več ljudi vgrajuje prezračevalne sisteme z rekuperacijo toplote, ki bistveno zmanjšajo izgube.

Večina prezračevalnih enot deluje nenehno, kar povzroča nepotrebne energetske izgube. Upravljanje je pogosto zastarelo, le redki sistemi omogočajo daljinsko upravljanje prek mobilne aplikacije. Centralizirane povezave otežujejo montažo v obstoječih objektih. Trenutni sistemi ne ponujajo celostnega profiliranja zraka in popolnoma avtonomnega delovanja. Prav tako so naprave povezane na domači usmerjevalnik, kar v določenih prostorih povzroča težave s signalom.

Projekt se osredotoča na lokalne prezračevalne enote, s poudarkom na optimizaciji geometrije in avtonomnem delovanju v sklopu razvoja lastnega sistema. Ta zajema programiranje mikrokrmilnikov za vodenje enote, razvoj mobilne aplikacije, CAD-modeliranje, izdelavo dokumentacije, 3D-tisk, načrtovanje vezij in simulacije pretoka zraka z predvidenimi izboljšavami:

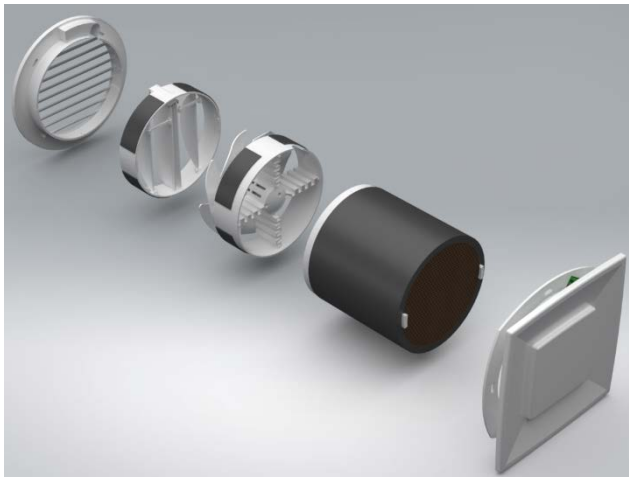
- Implementacija senzorjev za profiliranje zraka in obdelavo podatkov.
- Optimizacija ventilatorja s simulacijami pretoka za večje pretoke in manjšo hrupnost.
- Izboljšava električnega motorja in CAD modelov komponent.
- Omogočanje povezovanja naprav brez Wi-Fi in dodatnih modulov.
- Optimizacija toplotnega prenosnika glede na geometrijo in material.
- Načrtovanje algoritmov za optimalno delovanje glede na kakovost zraka.

2 Strojna oprema enote

Strojno opremo enote predstavlja krmilno vezje z ESP32 mikrokrmilnikom, ki ima integrirano periferijo za povezavo Wi-Fi, uporabniški vmesnik s tipkama in barvno LED diodo, za prikaz stanj. Ventilator poganja BLDC elektromotor z notranjim rotorjem, za pogon lopute pa je uporabljen DC elektromotor z reduktorjem.

Senzorski del predstavljata 2 Bosch BME680 senzorja, katera poganja napredna programska knjižnica z vgrajeno umetno inteligenco.

3 Optimizacija geometrije komponent



Slika 3.1: izgled prezračevalne enote po optimizaciji

3.1 Optimizacija ventilatorja

Ventilator je ključen za tiho in učinkovito delovanje enote. Prvotna izvedba je bila zasnovana izkustveno, kar je privedlo do slabih rezultatov. Simulacija v programu Ansys je pokazala neenakomeren pretok zraka, kar je povzročilo dodatne turbulence in nezadosten pretok 28 m3/h.

Zaradi teh težav smo analizirali različne tipe ventilatorjev ter njihove geometrijske lastnosti. Klasični ventilatorji z visokim pretokom niso bili ustrezni zaradi upada zmogljivosti pri oviranem pretoku. Zato smo se odločili za ventilator z visokim statičnim tlakom, saj enota vključuje keramični prenosnik toplote, ki predstavlja oviro.

Za začetno obliko smo analizirali enega najboljših ventilatorjev na trgu (Be Quiet! Silent

Wings 3). Ker naš ventilator potrebuje pretok v obe smeri, smo lopatice zasnovali simetrično, kar je zmanjšalo sicer zmogljivost, a omogočilo obojesmerno delovanje.

Modeliranje smo izvedli v programu SolidWorks z naprednimi funkcijami, kot so revolve in loft. Po začetni zasnovi smo iterativno izboljševali model s simulacijami v Ansysu. S spremembami geometrije lopatic smo izboljšali hitrost zraka in zmanjšali turbulence. Prelomna izboljšava je bila implementacija difuzorja na čelo pesta, kar je povečalo pretok za dodatnih 15 % in zmanjšalo šum. Po več iteracijah smo razvili končni model, ki pri 2000 vrtljajih na minuto zagotavlja 160 m3/h pretoka skozi enoto.

Ohišje smo oblikovali z namenom dušenja tresljajev. To nalogo opravlja lomljen nosilec pesta. Zaradi težav s pregrevanjem motorjev smo v pesto dodali hladilne reže in nadomestili penasto izolacijo z gumijasto pušo, ki omogoča neposredno hlajenje motorja skozi kovinsko ohišje. (glej sliko 3.1)

3.2 Oblikovanje pokrovov

Notranji pokrov smo preoblikovali za estetski videz in manjše dimenzije. Dodali smo ohišje za senzor, ki je zaščiten pred neposrednim zračnim tokom. Da bi preprečili napačne meritve zaradi mešanja zračne mase, smo simulirali pretok zraka in povečali globino pokrova ter s tem reže za boljši pretok.

Zunanji nadometni pokrov smo razvili na novo, saj je moral vključevati prostor za senzor. Zasnovan je tako, da odvaja vodo stran od prezračevalne cevi in preprečuje mešanje zunanjega in notranjega zraka. Za kupce, ki želijo diskreten videz, smo razvili tudi podometni pokrov. Ta se integrira v fasado, vključuje mrežo za preprečevanje razpok in omogoča nevidno vgradnjo. (glej sliko 3.1)

3.3 Razvoj lopute

Nova loputa je bila potrebna zaradi spremembe pokrova. Standardne lopute so prevelike za vgradnjo v tanke stene, zato smo razvili mehanizem s štirimi zaslonkami, zobniškim pogonom in vzvodi. To omogoča odpiranje vseh zaslonk z enim motorjem in vgradnjo v le 46 mm prostora, kar je znatno manj od klasičnih rešitev. (glej sliko 3.1)

4 Optimizacija prenosnika toplote ter določitev časa cikla z analizo toplotnih razmer v Matlab-u

Pri analizi konkurence smo ugotovili, da so časi vpihovanja in izpihovanja konstantni, kar lahko vodi v izgube pri velikih temperaturnih razlikah. Da bi optimizirali učinkovitost, smo razvili simulacijo toplotnega odziva keramičnega prenosnika toplote v Matlabu. Simulacija temelji na toplotnih enačbah [1] in upošteva ključne parametre, kot so masa prenosnika, specifična toplota materiala, površina kanalov, oblika in dimenzije kanalov ter hitrost zraka.

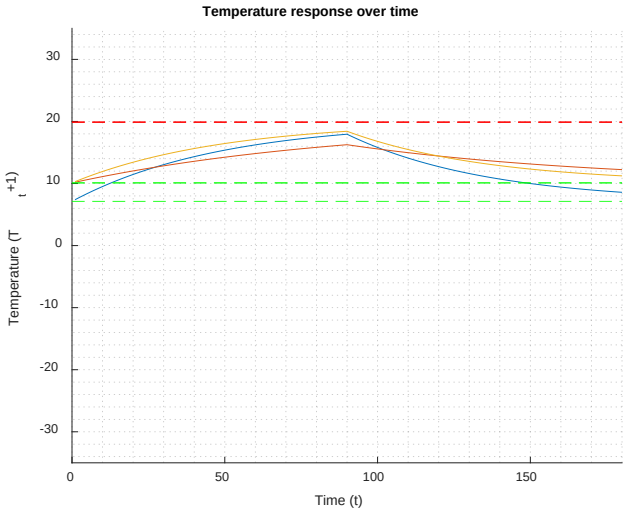
Najprej smo s simulacijo v Ansysu pridobili podatke o hitrosti zraka v kanalih pri različnih hitrostih ventilatorja ter jih vnesli v Matlab. Dodatno smo v vektorje zapisali lastnosti zraka pri temperaturah od -40 °C do 40 °C, kot so viskoznost, gostota in toplotna prevodnost. S temi podatki smo izračunali Reynoldsovo, Prandtlovo in Nusseltovo število ter koeficient prenosa toplote. Program je nato numerično izračunal temperaturo prenosnika s korakom ene sekunde, dokler razlika med notranjo temperaturo in prenosnikom ni dosegla minimalne vrednosti.

Optimizacijo smo nadgradili z algoritmom za izračun izkoristka, ki temelji na treh metodah: energijski bilanci prenosnika, učinkovitosti prenosa toplote pri izpihovanju in celotni učinkovitosti prenosa toplote v ciklu. Pri nizkih zunanjih temperaturah smo opazili, da prenosnik ob koncu cikla preveč ohlaja notranji prostor, zato smo prilagodili algoritem, da omeji ohlajanje ob večjih temperaturnih razlikah.

Dodatno smo analizirali vpliv geometrije prenosnika na izkoristek. Ugotovili smo, da manjši kanali z debelimi stenami povečajo čas polnjenja in praznjenja ter izboljšajo prestop toplote, a zmanjšajo pretok zraka. Sprva smo uporabljali kvadratne kanale velikosti 2–3 mm, nato pa prešli na 6-kotne, ki so omogočili večji izkoristek pri enakem pretoku. Pri optimizaciji

smo izvedli več iteracij, vključno z modeliranjem v SolidWorksu, simulacijami v Ansysu in analizami rezultatov v Matlabu. Končni prenosnik je po naših specifikacijah izdelalo podjetje Pingxiang Nanxiang Chemical Packing Co.

Grafi (glej sliko 4.1) prikazujejo toplotne odzive prenosnika pri različnih temperaturnih pogojih, geometrija s 6-kotnimi kanali, dosega boljše rezultate.



Slika 4.1: temperaturni odziv prenosnika toplote pri različnih pogojih

5 Zasnova programov in programiranje

Zaradi obsežnosti programa smo uporabili višje nivojski pristop objektno orientiranega programiranja, ki omogoča enkapsulacijo, dedovanje in polimorfizem. Pri izdelavi lastnih knjižnic smo uporabljali wrapper-je (slov. zavijalec). To je princip pisanja kode, kjer združimo več osnovnih metod v novi metodi, ki je običajno član nekega razreda.

5.1 Povezava in komunikacija

Knjižnica za povezavo in komunikacijo predstavlja hrbtenico sistema, saj omogoča krmiljenje enot in povratne informacije uporabniku. Implementacija izboljšane povezljivosti temelji na Wi-Fi protokolu, ki nudi večji doseg kot Bluetooth, energijska poraba pa ni kritična, saj sistem ni baterijsko napajan.

Za razvoj knjižnice comm_Utils smo uporabili štiri ključne knjižnice:

- WifiManager, ki omogoča prijavo in konfiguracijo omrežja (root node, mesh node, wifi direct node),
- Wifi, ki nudi podporo ostalim knjižnicam,
- PubSubClient, ki omogoča povezavo s strežnikom in prenos podatkov,
- PainlessMesh, ki skrbi za lastno mrežo naprav in avtomatsko posredovanje podatkov, tudi brez neposredne povezave z routerjem.

Uporabnik lahko določi napravo, ki se poveže v domače omrežje in deluje kot root node, ustvarja lastno mrežo (AP) ter povezuje druge naprave. Sistem omogoča:

- samodejno prepoznavanje root node-a in dodelitev pravic,
- dinamično iskanje najučinkovitejše poti za prenos podatkov brez posredovanja uporabnika,
- avtomatsko rekonfiguracijo mreže v primeru okvare posamezne naprave.

Ker knjižnice podpirajo le osnovne metode pošiljanja sporočil smo razvili lasten handling, ki omogoča pravilno usmerjanje sporočil glede na konfiguracijo in pravice naprav. Standardizirali smo format sporočil, ki zagotavlja, da lahko vse naprave komunicirajo s strežnikom neposredno ali posredno prek root node-a, stabilno povezavo ter minimalno obremenitev domačega omrežja

5.2 Implementacija BSEC2 knjižnice

BSEC2 knjižnica [2] podjetja Bosch Sensortec omogoča zajem in analizo okoljskih podatkov (temperatura, vlaga, tlak, kakovost zraka). Podpira različne senzorje, ponuja napredne algoritme in omogoča enostavno integracijo. Vendar ima eno ključno omejitev – omogoča uporabo pametnega algoritma le za en senzor hkrati, medtem ko naš sistem zahteva simultano obdelavo podatkov iz dveh senzorjev (zunanji in notranji).

Po poizvedbah so nam s strani Bosch Sensortec potrdili razvoj nove knjižnice, ki bo podpirala več senzorjev, vendar trenutno to še ni mogoče.

Ker nismo mogli čakati na uradno rešitev, smo knjižnico podvojili – ustvarili smo dve kopiji (bsec2_in in bsec2_out) ter preimenovali razrede, da ju je bilo mogoče sočasno uporabiti. Obe knjižnici smo vključili v lastno knjižnico air_data, ki omogoča simultano obdelavo podatkov iz obeh senzorjev. To nam je omogočilo pravilno delovanje sistema brez potrebe po dodatnih prilagoditvah originalne knjižnice.

5.3 obdelava in optimizacija vrednosti odčitkov kvalitete zraka

Knjižnica air_data združuje wrapper-je za bsec2_in in bsec2_out ter dodatne metode za obdelavo podatkov iz senzorjev. Algoritem za optimizacijo vrednosti smo prvotno razvili v Matlabu, nato pa ga prilagodili standardu C++.

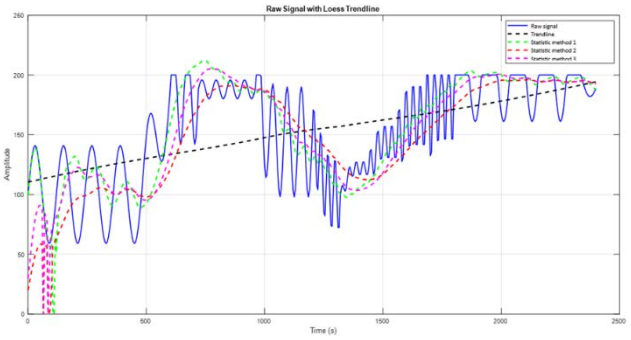
Problem nereprezentabilnih vrednosti kakovosti zraka – med uvodnim testiranjem smo se srečali s težavo nereprezentabilnih vrednosti kakovosti zraka, ki je posledica delovanja ventilatorja in velikega pretoka zraka. Ob izhodu skozi reže zunanjega ali notranjega pokrova zaradi visokih hitrosti nastaja turbulenca, ki povzroča mešanje izpuščenega zraka z zrakom v okolici ali notranjosti zgradbe.

V praksi to pomeni, da ko prezračevalna enota iz notranjosti črpa zrak proti okolici, se ta na izhodu meša z zunanjim zrakom. Zato senzorji ne zaznavajo prave kakovosti zunanjega zraka, temveč mešanico notranjega in zunanjega zraka. Podoben pojav se zgodi pri vpihovanju zraka iz okolice v notranjost, kjer se notranji zrak meša s svežim dovodnim zrakom, kar prav tako vpliva na odčitke senzorjev.

Ker se algoritem za avtonomno delovanje enote zanaša na te vrednosti, morajo biti podatki natančni in stabilni. Velike variacije ali odstopanja od trenda lahko povzročijo nestabilnost sistema. Zaradi tega smo morali najti rešitev za glajenje vrednosti in izboljšanje njihove uporabnosti v algoritmu za avtonomno delovanje.

Tako smo za izboljšanje natančnosti odčitkov preizkusili več statističnih metod za obdelavo signalov. Najprej smo začeli zbirati surove odčitke kakovosti zraka iz delujočih prototipov, skupaj s podatki o temperaturi in vlagi. Te vrednosti smo vpisovali v datoteke in jih analizirali po sedmih dneh zajemanja podatkov.

Po analizi smo ugotovili, da so določeni hitri skoki v podatkih posledica turbulentnega toka in ne dejanske spremembe kakovosti zraka. Zato smo za glajenje uporabili Lowess metodo, ki omogoča odstranjevanje kratkotrajnih nihanj, hkrati pa ohranja dolgoročne trende. Implementacijo smo prilagodili standardu C++ in jo vključili v knjižnico air_data, kjer je del algoritma za izboljšanje stabilnosti sistema. (glej sliko 5.1)



Slika 5.1: graf vhodnega signala ter izhodni signali različnih filtrov

Za preverjanje pravilnosti algoritma smo rezultate primerjali s tistimi iz Matlab-a, kar je potrdilo ustrezno implementacijo v C++. Tako smo zagotovili stabilne in natančne vrednosti kakovosti zraka, ki omogočajo pravilno delovanje prezračevalne enote brez neželenih nihanj ali nenadnih sprememb režima delovanja.

5.4 Izdelava algoritma za avtonomno delovanje

Algoritem za avtonomno delovanje predstavlja največjo konkurenčno prednost. Za svoje delovanje se zelo zanaša na vrednosti kvalitete zraka in izračune časa cikla. Z to podporo prezračevalni enoti algoritem omogoča, da sama prepoznava kvaliteto zraka tako v prostoru, kot zunaj in se ustrezno odziva. Sam algoritem vsebuje tudi varnostne ukrepe v primeru požara ali uhajanja plina in drugih strupenih substanc tako zunaj kot noter, kar poskrbi ne samo za boljšo kvaliteto bivanja ampak tudi za dodatno varnost.

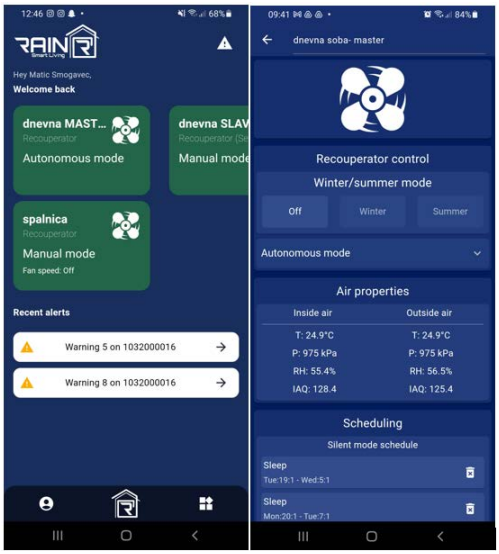
Sama logika algoritma je razdeljena na 2 nivoja, z upoštevanjem:

- Robnih (trdih) pogojev za delovanje
- Notranjih (mehkih) pogojev za delovanje
- Dodatnih pogojev

Robni pogoji, skrbijo oz. pokrivajo varnostno področje medtem, ko mehki pogoji skrbijo za čim večjo učinkovitost prezračevanja in pri tem še vedno ohranjajo kvaliteto bivanja (npr. ko je v prostoru več ljudi je zrak slabši in takrat prezračujemo z večjo močjo). Dodatni pogoji, pa omogočajo še druge funkcionalnosti, kot so načini gretja ali hlajenja

6 Izdelava aplikacije kot uporabniškega vmesnika

Upravljanje z napravo omogoča mobilna aplikacija, ki pošilja podatke na MQTT strežnik, ta pa nato komunicira z našimi napravami. MQTT je za naše naprave najprimernejši komunikacijski protokol, saj je bil zasnovan za izjemno lahek prenos sporočil po principu objavljanja in naročanja (publish and subscribe). Uporaben je za povezave z oddaljenimi lokacijami, kjer je prostor za programsko kodo omejen na nekaj kB in/ali pa je zelo pomembna čim manjša obremenitev omrežja. Pri izdelavi mobilne aplikacije (glej sliko 6.1) in programiranju serverja smo se zanesli na kolege iz FE UL. Sami pa smo poskrbeli za ustrezno upravljanje sporočil na samem mikrokrmilniku.



Slika 6.1: kolaž zaslonskih posnetkov mobilne aplikacije

7 Testiranje in odziv »beta tester«-jev

S predhodnimi izkušnjami smo mehanske komponente zasnovali z mislijo na zanesljivost. Največji izziv je bila loputa, vendar so bile potrebne izboljšave minimalne. Program smo testirali sproti, zato večjih napak ni bilo. Največjo negotovost je predstavljal avtonomni algoritem, saj se v praksi zaradi spremenljivih pogojev ni vedno obnesel. Testiranje je pokazalo, da se algoritem preveč burno odziva, zato smo vključili filter za obdelavo signalov.

Povezavo med enotami in strežnikom smo preizkusili že pri razvoju knjižnice comm_Utils. Prototipe smo testno vgradili v različnih mestih in na daljavo analizirali podatke ter prilagajali algoritme za stabilno delovanje. Uporabniki so sprva poročali o težavah, kot so izguba povezave, nestabilno delovanje in težave s krmiljenjem. Napake smo hitro odpravljali, kar je postopoma zmanjšalo negativne povratne informacije, katerih na koncu več ni bilo.

8 Literatura

[1] 2. Zrnić, S. J. Grejanje i klimatizacija. Beograd : Naučna knjiga, 1978.

[2] BME68X_SensorAPI. Github.com, 2024. Dostopno na: GitHub - boschsensortec/BME68x_SensorAPI: Common Sensor API for the BME680 and BME688 sensors [14. 10. 2024].

[3] MEMS sensors forum. Bosch-sensortec.com, 2024. Dostopno na: MEMS sensors forum (bosch-sensortec.com) [14. 10. 2024].